8-17-2021

Panoramic learning with a standardized machine learning formalism

Zhiting Hu
UC San Diego

Eric P. Xing
Carnegie Mellon University, United States & Mohamed bin Zayed University of Artificial Intelligence & Petuum Inc.

Preprint: arXiv;
Archived with thanks to arXiv;
Preprint License: CC by 4.0;
Uploaded 20 May 2022

Recommended Citation

Z. Hu and E. Xing, "Panoramic learning with a standardized machine learning formalism", arXiv, Aug. 2021, doi: 10.48550/arXiv.2108.07783

Panoramic Learning with A Standardized Machine Learning Formalism

Zhiting Hu[1], Eric P. Xing[2,3,4]

[1]UC San Diego, [2]Carnegie Mellon University, [3]MBZUAI, [4]Petuum Inc.

Abstract

Machine Learning (ML) is about computational methods that enable machines to learn concepts from experiences. In handling a wide variety of experiences ranging from data instances, knowledge, constraints, to rewards, adversaries, and lifelong interplay in an ever-growing spectrum of tasks, contemporary ML/AI research has resulted in a multitude of learning paradigms and methodologies. Despite the continual progresses on all different fronts, the disparate narrowly-focused methods also make standardized, composable, and reusable development of learning solutions difficult, and make it costly if possible to build AI agents that panoramically learn from all types of experiences. This paper presents a standardized ML formalism, in particular a standard equation of the learning objective, that offers a unifying understanding of diverse ML algorithms, making them special cases due to different choices of modeling components. The framework also provides guidance for mechanic design of new ML solutions, and serves as a promising vehicle towards panoramic learning with all experiences.

1 Introduction

Human learning has the hallmark of learning concepts from diverse sources of information. Take the example of learning a language. Humans can benefit from various experiences—by observing examples through reading and hearing, studying abstract definitions and grammar, making mistakes and getting correction from teacher, interacting with others and observing implicit feedback, etc. Knowledge of prior language can also accelerate the acquisition of new one. How can we build AI agents that are similarly capable of panoramically learning from all types of experiences?

In handling different experiences ranging from data instances, knowledge, constraints, to rewards, adversaries, and lifelong interplay in an ever-growing spectrum of tasks, contemporary ML and AI research has resulted in a large multitude of learning paradigms (e.g., supervised, unsupervised, active, reinforcement, adversarial learning), models, optimization techniques, not mentioning countless approximation heuristics and tuning tricks, plus combinations of all above. While pushing the field forward rapidly, these results also make mastering existing ML techniques very difficult, and fall short of reusable, repeatable, and composable development of ML solutions to diverse problems with distinct available experiences.

The principal goal in this paper is to present a standardized ML formalism that offers a principled framework for understanding, unifying, and generalizing current major paradigms of learning algorithms, and for mechanic design of new solutions for integrating any useful experiences in learning. The power of standardized theory is perhaps best demonstrated in Physics which has a long history of chasing symmetry and simplicity of its principles: exemplified by the famed Maxwell's equations in 1800s which reduced various principles of electricity and magnetism into a single electromagnetic theory, followed by General Relativity in 1910s and the Standard Model in 1970s, physicists describe the world best by unifying and reducing different theories to a standardized one. Likewise, it is a constant quest in the field of ML to establish a "Standard Model" (Langley, 1989; Domingos, 2015), that gives a holistic view of the broad learning principles, lays out a blueprint permitting fuller and more systematic exploration in the design and analysis of new algorithms, and eventually serves as a vehicle towards panoramic learning that integrates all available sources of experiences. This paper presents an attempt towards this end.

We investigate the underlying connections between a range of seemingly distinct ML paradigms. Each of these paradigms has made particular assumptions on the form of experiences available. For example, the present most popular *supervised learning* relies on collections of data instances, often applying a maximum likelihood objective solved with simple gradient descent. Maximum likelihood based *unsupervised learning* instead can invoke different solvers, such as expectation-maximization (EM), variational inference, and wake-sleep in training for varied degree of approximation to the problem. *Active learning* (Settles, 2012) manages data instances which, instead of being given all at once, are adaptively selected. *Reinforcement learning* (Sutton and Barto, 2017) makes use of feedback obtained via interaction with the environment. *Knowledge-constrained learning* like posterior regularization (Ganchev et al., 2010; Zhu et al., 2014) incorporates structures, knowledge, and rules expressed as constraints. *Generative adversarial learning* (Goodfellow et al., 2014) leverages a companion model called discriminator to guide training of the model of interest.

The standardized ML formalism is materialized as a standard equation (SE) of the objective function that drives the model training given any experiences. The SE formulates a rather broad design space of learning algorithms. We show that many of the well-known algorithms of the above paradigms are all instantiations of the general formulation. More specifically, the SE, based on the maximum entropy and variational principles, consists of three principled terms, including the *experience* term that offers a unified language to express arbitrary relevant information to supervise the learning, the *divergence* term that measures the fitness of the target model to be learned, and the *uncertainty* term that regularizes the complexity of the system. The single succinct formula re-derives the objective functions of myriad learning algorithms, reducing them to different choices of the components. The formulation thus shed new lights on the fundamental relationships between the diverse algorithms that were each originally designed to deal with a specific type of experience.

The modularity and generality of the framework is particularly appealing not only from the theoretical point of view, but also because it offers guiding principles for designing algorithmic solutions to new problems in a mechanic way. Specifically, the SE is naturally designed to allow combining together all different experiences to learn a model of interest. Designing a problem solution boils down to choosing *what* experiences to use depending on the problem structure and available resources, without worrying too much about *how* to use the experiences in the training. The standardized ML perspective also highlights that many learning problems in different research areas are essentially the same and just correspond to specifications of the SE components. This enables us to systematically re-purpose successful techniques in one area to solve problems in another.

The remainder of the paper is organized as follows. Section 2 gives an overview of relevant learning and inference techniques as a prelude of the standardized framework. Section 3 presents the standard equation as a general formulation of learning algorithms. The subsequent two sections discuss different choices of the two key components in the standard equation, respectively, illustrating that many existing methods are special cases of the formulation: Section 4 is devoted to discussion of the experience function and Section 5 focuses on the divergence measure. Section 6 discusses the solving algorithms for optimizing the standard equation objective. Section 7 discusses the utility of the standardized formalism for mechanic design of panoramic learning solutions. Section 8 reviews related work. Section 9 concludes the paper with discussion of some of the future directions.

2 Preliminaries: The Maximum Entropy View of Learning and Inference

Depending on the nature of the task (e.g., classification or regression), data (e.g., labeled or unlabeled), information scope (e.g., with or without latent variables), and form of domain knowledge (e.g., prior distributions or parameter constraints), etc., different learning paradigms with often complementary (but not necessarily inclusive) advantages have been developed for different needs. For example, the paradigms built on the maximum likelihood principles, Bayesian theories, variational calculus, and Monte Carlo simulation have led to much of the foundation underlying a wide spectrum of probabilistic graphical models, exact/approximate inference algorithms, and even probabilistic logic programs suitable for probabilistic inference and parameter estimations in multivariate, structured, and fully or partially observed domains. Whereas the paradigms built on convex optimization, duality theory, regularization, and risk minimization have led to much of the foundation underlying algorithms such as SVM, boosting, sparse learning, structure learning, etc. Historically, there has been numerous efforts in establishing a unified machine learning framework that can bridge these complementary paradigms so that advantages in model design, solver efficiency, side-information incorporation, and theoretical guarantees can be translated across paradigms. As a prelude of our presentation of the "standard equation" framework toward this goal, here we begin with a recapitulation of the maximum entropy view of statistical learning. By naturally marrying the probabilistic frameworks with the optimization-theoretic frameworks, the maximum entropy viewpoint had played an important historical role in offering the same lens to understanding several popular methodologies such as maximum likelihood learning, Bayesian inference, and large margin learning.

2.1 MLE

2.1.1 Supervised MLE

Denote the model to be learned as $p_\theta(\boldsymbol{x})$, where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is the model parameters. The most common method for estimating the parameters $\boldsymbol{\theta}$ is perhaps maximum likelihood estimation (MLE). Given a set of fully-observed data examples $\mathcal{D} = \{\boldsymbol{x}^*\}$, MLE learns the model by minimizing the negative log-likelihood:

$$\min_{\boldsymbol{\theta}} -\mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} \left[\log p_\theta(\boldsymbol{x}^*)\right]. \tag{2.1}$$

MLE is known to be intimately related to the maximum entropy principle (Jaynes, 1957). In particular, when the model $p_\theta(\boldsymbol{x})$ is in the exponential family of the form:

$$p_\theta(\boldsymbol{x}) = \exp\left\{\boldsymbol{\theta} \cdot T(\boldsymbol{x})\right\} / Z(\boldsymbol{\theta}), \tag{2.2}$$

where $T(\boldsymbol{x})$ is the features of data $\boldsymbol{x}$ and $Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \exp\{\boldsymbol{\theta} \cdot T(\boldsymbol{x})\}$ is the normalization factor, it is shown that MLE is the convex dual of maximum entropy estimation.

In the maximum entropy formulation, rather than assuming a specific parametric from of the target model distribution, denoted as $p(\boldsymbol{x})$, we instead impose constraints on the model distribution, which requires the expectation of the features $T(\boldsymbol{x})$ to be equal to the empirical expectation:

$$\mathbb{E}_p\left[T(\boldsymbol{x})\right] = \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}}\left[T(\boldsymbol{x}^*)\right]. \tag{2.3}$$

Let $\mathcal{P}(\mathcal{X})$ denote the set of all probability distributions on $\mathcal{X}$. In general, there exist many distributions $p \in \mathcal{P}(\mathcal{X})$ that satisfy the constraint. The principle of maximum entropy resolves the ambiguity by choosing the distribution such that its Shannon entropy, $\mathrm{H}(p) := -\mathbb{E}_p[\log p(\boldsymbol{x})]$, is maximized. We thus have the constrained optimization problem:

$$\begin{aligned} \min_{p(\boldsymbol{x})} \;\; & \mathrm{H}\left(p(\boldsymbol{x})\right) \\ s.t. \;\; & \mathbb{E}_p\left[T(\boldsymbol{x})\right] = \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}}\left[T(\boldsymbol{x}^*)\right] \\ & p(\boldsymbol{x}) \in \mathcal{P}(\mathcal{X}). \end{aligned} \tag{2.4}$$

The problem can be solved with the Lagrangian method. Specifically, we write the Lagrangian:

$$\mathcal{L}(p, \boldsymbol{\theta}, \mu) = \mathrm{H}(p(\boldsymbol{x})) - \boldsymbol{\theta}\left(\mathbb{E}_p\left[T(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}}\left[T(\boldsymbol{x}^*)\right]\right) - \mu\left(\sum\nolimits_{\boldsymbol{x}} p(\boldsymbol{x}) - 1\right), \tag{2.5}$$

where $\boldsymbol{\theta}$ and μ are Lagrangian multipliers. Minimizing w.r.t p leads to the optimal solution which turns out to be in the same form of Eq.(2.6):

$$p(\boldsymbol{x}) = \exp\left\{\boldsymbol{\theta} \cdot T(\boldsymbol{x})\right\} / Z(\boldsymbol{\theta}), \tag{2.6}$$

where we see the parameters $\boldsymbol{\theta}$ in the exponential family parameterization are the Lagrangian multipliers that enforce the constraints. Plugging the solution back into the Lagrangian, we obtain:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}}\left[\boldsymbol{\theta} \cdot T(\boldsymbol{x}^*)\right] - \log Z(\boldsymbol{\theta}), \tag{2.7}$$

which is simply the negative of the MLE objective in Eq.(2.1).

Thus maximum entropy is dual to maximum likelihood. It provides an alternative view of the problem of fitting a model into data, where the data instances in the training set are treated as constraints, and the learning problem is treated as a constrained optimization problem. This optimization-theoretic view of learning will be re-visited repeatedly in the sequel to allow extending machine learning under all experiences of which data instances is just a special case.

Iterative Proportional Fitting (IPF) The optimization-theoretic view of learning offered by the maximum entropy formulation not only allows to accommodate other forms of experiences beyond data as we will show in the sequel, but also provides a potentially standardized paradigm to devise efficient solvers for the learning problem, as we will explore in this paper. As an example, consider the MLE problem of undirected graphical models (Jordan, 2003) whose distribution is generally expressed as:

$$p_\theta(\boldsymbol{x}) = \prod\nolimits_{c \in \mathcal{C}} \exp\{\psi_c(\boldsymbol{x}_c)\}/Z, \tag{2.8}$$

where $\mathcal{C}$ is a set of cliques in the graph; $\boldsymbol{x}_c$ is the variables associated with the clique $c \in \mathcal{C}$; $\exp\{\psi_c(\boldsymbol{x}_c)\}$ is a clique potential for the clique c; $\boldsymbol{\theta} = \{\psi_c(\boldsymbol{x}_c) : c \in \mathcal{C}\}$ is the collection of parameters to be learned; and $Z = \sum_{\boldsymbol{x}} \prod_c \exp\{\psi_c(\boldsymbol{x}_c)\}$ is the normalization factor.

To find the maximum likelihood estimates, direct gradient descent on the negative log-likelihood (Eq.2.1) is inefficient due to the cumbersome $\log Z$ term. Iterative Proportional Fitting (IPF, Deming and Stephan, 1940) is a generic solver for the estimation problem, which can readily be derived from the maximum entropy formulation. Specifically, let N be the size of the observed dataset $\mathcal{D}$, $m(\boldsymbol{x})$ the number of times that configuration $\boldsymbol{x}$ is observed in $\mathcal{D}$, and $m(\boldsymbol{x}_c) = \sum_{x_{\setminus c}} m(\boldsymbol{x})$ the marginal count for clique c by summing (or integrating in the continuous case) over all configurations of variables $\boldsymbol{x}_{\setminus c}$ not included in the clique c. Let $\tilde{p}_d(\boldsymbol{x}_c) = m(\boldsymbol{x}_c)/N$ be the empirical marginal. The dual maximum entropy formulation of the MLE is then written as:

$$\begin{aligned} \min_{p(\boldsymbol{x})} \ & \mathrm{H}\left(p(\boldsymbol{x})\right) \\ s.t. \ & p(\boldsymbol{x}_c) = \tilde{p}_d(\boldsymbol{x}_c), \ \forall c \in \mathcal{C} \\ & p(\boldsymbol{x}) \in \mathcal{P}(\mathcal{X}), \end{aligned} \tag{2.9}$$

where the constraints are that the model marginals $p(\boldsymbol{x}_c)$ must be equal to the empirical marginals $\tilde{p}_d(\boldsymbol{x}_c)$ for each clique c. It is worth noting that the marginal constraint is a special case of the expected feature constraint in Eq.(2.3), by using features that are indicator functions. That is, for a configuration $\boldsymbol{x}_c^*$, the marginal probability can be viewed as the expectation $p(\boldsymbol{x}_c^*) = \mathbb{E}_{p(\boldsymbol{x})}[\mathbb{I}_{x_c^*}(\boldsymbol{x}_c)]$, where the feature $\mathbb{I}_{x_c^*}(\boldsymbol{x}_c)$ is an indicator function that equals 1 when $\boldsymbol{x}_c = \boldsymbol{x}_c^*$ and 0 otherwise.

Again, we solve the problem by introducing the Lagrangian multipliers $\psi_c(\boldsymbol{x}_c)$ to impose the marginal constraints and μ for the normalization constraint, resulting in the Lagrangian:

$$\mathcal{L}(p, \{\psi_c(\boldsymbol{x}_c)\}, \mu) = \mathrm{H}(p(\boldsymbol{x})) - \sum_{c, \boldsymbol{x}_c} \psi_c(\boldsymbol{x}_c)\Big(p(\boldsymbol{x}_c) - \tilde{p}_d(\boldsymbol{x}_c)\Big) - \mu\left(\sum_{\boldsymbol{x}} p(\boldsymbol{x}) - 1\right). \tag{2.10}$$

Solving the Lagrangian gives the following update w.r.t $\psi_c(\boldsymbol{x}_c)$ (see Teh and Welling (2003) for derivations) at each iteration n:

$$\psi_c^{(n+1)}(\boldsymbol{x}_c) = \psi_c^{(n)}(\boldsymbol{x}_c) + \log \frac{\tilde{p}_d(\boldsymbol{x}_c)}{p^{(n)}(\boldsymbol{x}_c)}. \tag{2.11}$$

which is the classical form of IPF updates if we see the correspondence between the Lagrangian multipliers $\psi_c(\boldsymbol{x}_c)$ and the clique potentials. The update is equivalent to updating the primal variables $p(\boldsymbol{x})$ with:

$$p^{(n+1)}(\boldsymbol{x}) = p^{(n)}(\boldsymbol{x}) \frac{\tilde{p}_d(\boldsymbol{x}_c)}{p^{(n)}(\boldsymbol{x}_c)}. \tag{2.12}$$

The IPF procedure can be seen as iterating over the cliques in $\mathcal{C}$ and for each clique c setting the marginal $p(\boldsymbol{x}_c)$ to be $\tilde{p}_d(\boldsymbol{x}_c)$.

As we will show and continue to extend in the sequel, the alternating minimization procedure over a Lagrangian that facilitates arbitrary combination of loss and constraints, as implemented by the IPF, represents a general idea behind a turnkey solver for learning systems defined by potentially a general objective function that can subsume a wide spectrum of ML paradigms, including those in the past appeared to rely heavily on specially developed solvers requiring substantial mathematical insights and incurring substantial computational complexity. With different choice of the objective functions such as over different forms of experiences and losses, minimization techniques such as gradient, Langevin methods, we can recover all known algorithms used for different ML problems, but also extrapolate into new models and algorithms with little efforts.

2.1.2 Unsupervised MLE

Similar to the MLE framework for supervised learning, unsupervised learning via MLE can also be reformulated as an constraint optimization problem via the maximum entropy principle. Consider learning a multivariate model with latent variables, where each data instance is partitioned into observed variables $\boldsymbol{x}$ and latent variables $\boldsymbol{y}$. The goal is to learn a model $p_\theta(\boldsymbol{x}, \boldsymbol{y})$ that captures the joint distribution of $\boldsymbol{x}$ and $\boldsymbol{y}$. Since $\boldsymbol{y}$ is unobserved, we minimize the negative log-likelihood with $\boldsymbol{y}$ marginalized out:

$$\min_{\boldsymbol{\theta}} -\mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} \left[\log \sum_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \right]. \tag{2.13}$$

Direct optimization of the marginal log-likelihood is typically intractable due to the integration over $\boldsymbol{y}$. Earlier work thus developed different solvers with varying levels of approximations.

It can be shown that the intractable negative log-likelihood above can be upper bounded by a more tractable term known as the *variational free energy* (Neal and Hinton, 1998). Let $q(\boldsymbol{y}|\boldsymbol{x})$ represent an arbitrary auxiliary distribution acting as a surrogate of the true posterior $p(\boldsymbol{y}|\boldsymbol{x})$, which is known as a variational distribution. Then, for each instance $\boldsymbol{x}^* \in \mathcal{D}$, we have:

$$\begin{aligned} -\log \sum_{\boldsymbol{y}} p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) &= -\mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)} \left[\log \frac{p_\theta(\boldsymbol{x}^*, \boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x}^*)} \right] - \mathrm{KL}\left(q(\boldsymbol{y}|\boldsymbol{x}^*) \| p_\theta(\boldsymbol{y}|\boldsymbol{x}^*) \right) \\ &\leq -\mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)} \left[\log \frac{p_\theta(\boldsymbol{x}^*, \boldsymbol{y})}{q(\boldsymbol{y}|\boldsymbol{x}^*)} \right] \\ &= -\mathrm{H}\left(q(\boldsymbol{y}|\boldsymbol{x}^*) \right) - \mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)} \left[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \right] := \mathcal{L}(q, \boldsymbol{\theta}), \end{aligned} \tag{2.14}$$

where the inequality holds because KL divergence is always non-negative. The free energy upper bound contains two terms: the first one is the entropy of the variational distribution, capturing intrinsic randomness (i.e., amount of information carried by an auxiliary distribution); the second term, now written as $-\mathbb{E}_{q(\boldsymbol{y}|\boldsymbol{x}^*)\tilde{p}_d(\boldsymbol{x}^*)} \left[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \right]$, by taking into account the empirical distribution $\tilde{p}_d$ from which the instance $\boldsymbol{x}^*$ is drawn, is the cross entropy between the distributions $q(\boldsymbol{y}|\boldsymbol{x}^*)\tilde{p}_d(\boldsymbol{x}^*)$ and $p_\theta(\boldsymbol{x}^*, \boldsymbol{y})$, driving the two to be close and thereby allowing q to approximate p.

The popular Expectation Maximization (EM) algorithm for unsupervised learning via MLE can be interpreted as minimizing the variational free energy (Neal and Hinton, 1998). In fact, as we discuss subsequently, popular heuristics such as the variational EM and the wake-sleep algorithms, are approximations to the EM algorithm by introducing approximating realizations to either the free energy objection function $\mathcal{L}$ or to the solution space of the variational distribution q.

Expectation Maximization (EM) The most common approach to learning with unlabeled data or partially observed multivariate models is perhaps the expectation-maximization (EM) algorithm (Dempster et al., 1977). With the use of the variational free energy as a surrogate objective to the original marginal likelihood as in Eq.(2.14), similar to the IPF algorithm described above, EM can be also understood as an alternating minimization algorithm, where $\mathcal{L}(q, \boldsymbol{\theta})$ is minimized w.r.t q and $\boldsymbol{\theta}$ in two stages, respectively. At each iteration n, the expectation (E) step maximizes $\mathcal{L}(q, \boldsymbol{\theta}^{(n)})$ w.r.t q. From Eq.(2.14), this is achieved by setting q to the current true posterior:

$$\text{E-step:} \quad q^{(n+1)}(\boldsymbol{y}|\boldsymbol{x}^*) = p_{\theta^{(n)}}(\boldsymbol{y}|\boldsymbol{x}^*), \tag{2.15}$$

so that the KL divergence vanishes and the upper bound is tight. In the subsequent maximization (M) step, $\mathcal{L}(q^{(n+1)}, \boldsymbol{\theta})$ is minimized w.r.t $\boldsymbol{\theta}$:

$$\text{M-step:} \quad \max_{\boldsymbol{\theta}} \mathbb{E}_{q^{(n+1)}(\boldsymbol{y}|\boldsymbol{x}^*)} \left[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y}) \right], \tag{2.16}$$

which is to maximize the expected complete data log-likelihood. The EM algorithm has an appealing property that it monotonically decreases the negative marginal log-likelihood over iterations. To see this, notice that after the E-step the upper bound $\mathcal{L}(q^{(n+1)}, \boldsymbol{\theta}^{(n)})$ is equal to the negative marginal log-likelihood, and the M-step further decreases the upper bound (and thus the negative marginal log-likelihood).

Variational EM When the model $p_\theta(\boldsymbol{x}, \boldsymbol{y})$ is complex (e.g., a neural network or a multi-layer graphical model), directly working with the true posterior in the E-step becomes intractable. Variational EM overcomes the difficulty with approximations. The approach considers a restricted family $\mathcal{Q}'$ of the variational distribution $q(\boldsymbol{y})$ such that optimization w.r.t q within the family is tractable:

$$\text{Variational E-step:} \quad \min_{q \in \mathcal{Q}'} \mathcal{L}(q, \boldsymbol{\theta}^{(t)}). \tag{2.17}$$

A common way to restrict the q family is the <u>mean-field methods</u> which partition the components of $\boldsymbol{y}$ into sub-groups $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M)$ and assume that q factorizes w.r.t the groups: $q(\boldsymbol{y}) = \prod_{i=1}^{M} q_i(\boldsymbol{y}_i)$. The variational principle summarized in (Wainwright and Jordan, 2008) gives a more principled interpretation of the mean-field and other approximation methods. In particular, in the case where $p_\theta(\boldsymbol{x}, \boldsymbol{y})$ is an exponential family distribution with sufficient statistics $T(\boldsymbol{x}, \boldsymbol{y})$, the exact E-step (Eq.2.15) can be interpreted as seeking the optimal valid <u>mean parameters</u> (i.e., expected sufficient statistics) for which the free energy is minimized. For discrete latent variables $\boldsymbol{y}$, the set of all valid mean parameters constitutes a marginal polytope $\mathcal{M}$. In this perspective, the mean-field methods (Eq.2.17) correspond to replacing $\mathcal{M}$ with an <u>inner</u> approximation $\mathcal{M}' \subseteq \mathcal{M}$. With the restricted set $\mathcal{M}'$ of mean parameters, the E-step generally no longer tightens the bound of the negative marginal log-likelihood, and the algorithm does not necessarily decrease the negative marginal log-likelihood monotonically. However, the algorithm preserves the property that it minimizes the upper bound of the negative marginal log-likelihood. Besides the mean-field methods, there are other approaches for approximation such as belief propagation. These methods correspond to using an <u>outer</u> approximation $\mathcal{M}'' \supseteq \mathcal{M}$ of the marginal polytope, and do not guarantee upper bounds on the negative marginal log-likelihood.

Another approach to restrict the family of q is to assume a parametric distribution $q_\omega(\boldsymbol{y}|\boldsymbol{x})$ and optimize the parameters $\boldsymbol{\omega}$ in the E-step. The approach has been used in black-box variational inference (Ranganath et al., 2014), and variational auto-encoders (VAEs) (Kingma and Welling, 2014) where q is parameterized as a neural network (a.k.a "inference network", or "encoder").

Wake-Sleep In some cases when the auxiliary q is assumed to have a certain form (e.g., a deep network), the approximate E-step in Eq.(2.17) may still be too complex to be tractable, or the gradient estimator (w.r.t the parameters of q) can suffer from high variance (Paisley et al., 2012; Mnih and Gregor, 2014). To tackle the challenge, more approximations are introduced. Wake-sleep algorithm (Hinton et al., 1995) is one of such methods. In the E-step w.r.t q, rather than minimizing $\text{KL}(q(\boldsymbol{y}) \| p_\theta(\boldsymbol{y}|\boldsymbol{x}^*))$ (Eq.2.14) as in EM and variational EM, the wake-sleep algorithm makes an approximation by minimizing the KL divergence in opposite direction:

$$\text{Approximate E-step (Sleep-phase):} \quad \min_{q \in \mathcal{Q}'} \text{KL}\left(p_\theta(\boldsymbol{y}|\boldsymbol{x}^*) \| q(\boldsymbol{y})\right), \tag{2.18}$$

which can be optimized efficiently with gradient descent when q is parameterized. Besides wake-sleep, one can also use other methods for low-variance gradient estimation in Eq.(2.17), such as reparameterization gradient (Kingma and Welling, 2014) and score gradient (Glynn, 1990; Ranganath et al., 2014; Mnih and Gregor, 2014).

In sum, the maximum entropy perspective has formulated unsupervised MLE as an optimization-theoretic framework that permits simple alternating minimization solvers. Starting from the upper bound of negative marginal log-likelihood (Eq.2.14) with maximum entropy and minimum cross entropy, the originally intractable MLE problem gets simplified, and a series of solving algorithms, ranging from (variational) EM to wake-sleep, arise naturally as an approximation to the original solution.

2.2 Bayesian Inference

Now we revisit another classical learning framework, Bayesian inference, and examine its intriguing connections with the maximum entropy principle. Interestingly, the the maximum entropy principle can also help to reformulate Bayesian inference as a constraint optimization problem, as for MLE.

Different from MLE, Bayesian approach for statistical inference treats the hypotheses (parameters $\boldsymbol{\theta}$) to be inferred as random variables. Assuming a prior distribution $\pi(\boldsymbol{\theta})$ over the parameters and considering the model to be a conditional distribution $p(\boldsymbol{x}|\boldsymbol{\theta})$, the inference is based on the Bayes's theorem:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{\pi(\boldsymbol{\theta}) \prod_{\boldsymbol{x}^* \in \mathcal{D}} p(\boldsymbol{x}^*|\boldsymbol{\theta})}{p(\mathcal{D})}, \tag{2.19}$$

where $p(\boldsymbol{\theta}|\mathcal{D})$ is the posterior distribution after observing the data $\mathcal{D}$; and $p(\mathcal{D}) = \int_{\boldsymbol{\theta}} \pi(\boldsymbol{\theta}) \prod_{\boldsymbol{x}^*} p(\boldsymbol{x}^*|\boldsymbol{\theta})$ is the marginal likelihood.

Interestingly, the early work by Zellner (1988) showed the relations between Bayesian inference and maximum entropy, by re-formulating the statistical inference problem from the perspective of information processing, and re-discovering the Bayes' theorem as the optimal information processing rule. More specifically, statistical inference can be seen as a procedure of information processing, where the system receives input information in the form of prior knowledge and data, and emits output information in the form of parameter estimates and others. An efficient inference procedure should generate an output distribution such that the system retains all input information and not inject any extraneous information. The learning objective is thus to minimize the difference between the input and output information w.r.t the output distribution:

$$\begin{aligned} \min_{q(\boldsymbol{\theta})} \quad & -\mathrm{H}(q(\boldsymbol{\theta})) + \log p(\mathcal{D}) - \mathbb{E}_{q(\boldsymbol{\theta})}\left[\log \pi(\boldsymbol{\theta}) + \sum\nolimits_{\boldsymbol{x}^* \in \mathcal{D}} \log p(\boldsymbol{x}^*|\boldsymbol{\theta})\right] \\ s.t. \quad & q(\boldsymbol{\theta}) \in \mathcal{P}(\boldsymbol{\Theta}), \end{aligned} \tag{2.20}$$

where the first two terms measure the output information in the output distribution $q(\boldsymbol{\theta})$ and marginal $p(\mathcal{D})$, and the third term measures the input information in the prior $\pi(\boldsymbol{\theta})$ and data likelihood $p(\boldsymbol{x}^*|\boldsymbol{\theta})$. Here $\mathcal{P}(\boldsymbol{\Theta})$ is the space of all probability distributions over $\boldsymbol{\theta}$.

The optimal solution of $q(\boldsymbol{\theta})$ is precisely the the posterior distribution $p(\boldsymbol{\theta}|\mathcal{D})$ due to the Bayes' theorem (Eq.2.19). The proof is straightforward by noticing that the objective can be further rewritten as $\min_q \mathrm{KL}(q(\boldsymbol{\theta})\|p(\boldsymbol{\theta}|\mathcal{D}))$.

Similar to the case of duality between MLE and maximum entropy (Eq.2.4), the same entropy maximum principle can cast Bayesian inference as a constrained optimization problem. As Jaynes (1988) commented, this fresh interpretation of Bayes' theorem "could make the use of Bayesian methods more attractive and widespread, and stimulate new developments in the general theory of inference". The next subsection reviews how entropy maximization as a "useful tool in generating probability distributions" (Jaynes, 1988) has related to and resulted in more general learning and inference frameworks, such as posterior regularization.

2.3 Posterior Regularization

The optimization-based formulation of Bayesian inference in Eq.(2.20) offers important additional flexibility in learning by allowing rich constraints on machine learning models to be imposed to regularize the outcome. For example, in Eq.(2.20) we have seen the standard normality constraint of a probability distribution being imposed on the posterior q. It is natural to consider other types of constraints that encode richer problem structures and domain knowledge, which can regularize the model to learn desired behaviors.

The idea has led to posterior regularization (PR, Ganchev et al., 2010) or regularized Bayes (RegBayes, Zhu et al., 2014) which augment the Bayesian inference objective with additional constraints:

$$\begin{aligned} \min_{q, \boldsymbol{\xi}} \quad & -\mathrm{H}\left(q(\boldsymbol{\theta})\right) - \mathbb{E}_{q(\boldsymbol{\theta})}\left[\sum\nolimits_{\boldsymbol{x}^* \in \mathcal{D}} \log p(\boldsymbol{x}^*|\boldsymbol{\theta})\pi(\boldsymbol{\theta})\right] + U(\boldsymbol{\xi}) \\ s.t. \quad & q(\boldsymbol{\theta}) \in \mathcal{Q}(\boldsymbol{\xi}) \\ & \boldsymbol{\xi} \geq 0, \end{aligned} \tag{2.21}$$

where we have rearranged the terms and dropped any constant factors in Eq.(2.20), and added constraints with $\boldsymbol{\xi}$ being a vector of slack variables, $U(\boldsymbol{\xi})$ a penalty function (e.g., ℓ_1 norm of $\boldsymbol{\xi}$), and $\mathcal{Q}(\boldsymbol{\xi})$ a subset of valid distributions over $\boldsymbol{\theta}$ that satisfy the constraints determined by $\boldsymbol{\xi}$. The optimization problem is generally easy to solve when the penalty/constraints are convex and defined w.r.t a linear operator (e.g., expectation) of the posterior q. For example, let $T(\boldsymbol{x}^*; \boldsymbol{\theta})$ be a feature vector of data instance $\boldsymbol{x}^* \in \mathcal{D}$, the constraint posterior set Q can be defined as:

$$Q(\boldsymbol{\xi}) := \left\{q(\boldsymbol{\theta}) \;:\; \mathbb{E}_q\left[T(\boldsymbol{x}^*; \boldsymbol{\theta})\right] \leq \boldsymbol{\xi}\right\}, \tag{2.22}$$

which bounds the feature expectations with $\boldsymbol{\xi}$.

Max-margin constraint is another expectation constraint that has shown to be widely effective in classification and regression (Vapnik, 1998). The maximum entropy discrimination (MED) by

Jaakkola et al. (2000) regularizes linear regression models with the max-margin constraints, which is latter generalized to more complex models $p(\boldsymbol{x}|\boldsymbol{\theta})$, such as Markov networks (Taskar et al., 2004) and latent variable models (Zhu et al., 2014). Formally, let $\boldsymbol{y}^* \in \mathbb{R}$ be the observed label associated with $\boldsymbol{x}^*$. The margin-based constraint says that a classification/regression function $h(\boldsymbol{x};\boldsymbol{\theta})$ should make at most ϵ deviation from the true label $\boldsymbol{y}^*$. Specifically, consider the common choice of the function h as a linear function: $h(\boldsymbol{x};\boldsymbol{\theta}) = \boldsymbol{\theta}^\top T(\boldsymbol{x})$, where $T(\boldsymbol{x})$ is, with a slight abuse of notation, the feature of instance $\boldsymbol{x}$. The constraint is written as:

$$\begin{cases} \boldsymbol{y}^* - \mathbb{E}_q\left[\boldsymbol{\theta}^\top T(\boldsymbol{x}^*)\right] \leq \epsilon + \xi \\ -\boldsymbol{y}^* + \mathbb{E}_q\left[\boldsymbol{\theta}^\top T(\boldsymbol{x}^*)\right] \leq \epsilon + \xi', \end{cases} \tag{2.23}$$

for all instances $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in \mathcal{D}$.

Alternating optimization for posterior regularization Having seen EM-style alternating minimization algorithms being applied as a general solver for a number of optimization-theoretic frameworks described above, it is not surprising that the posterior regularization framework can also be solved with an alternating minimization procedure. For example, consider the simple case of linear constraint in Eq.(2.22), penalty function $U(\boldsymbol{\xi}) = \|\boldsymbol{\xi}\|_1$, and q factorizing across $\boldsymbol{\theta} = \{\boldsymbol{\theta}_c\}$. At each iteration n, the solution of $q(\boldsymbol{\theta}_c)$ is given as:

$$q^{(n+1)}(\boldsymbol{\theta}_c) = \exp\left\{\mathbb{E}_{q^{(n)}(\boldsymbol{\theta}_{\backslash c})} \sum_{\boldsymbol{x}^*} \log p(\boldsymbol{x}^*|\boldsymbol{\theta})\pi(\boldsymbol{\theta}) + T(\boldsymbol{x}^*;\boldsymbol{\theta})\right\}/Z, \tag{2.24}$$

where $\boldsymbol{\theta}_{\backslash c}$ denotes all components of $\boldsymbol{\theta}$ except $\boldsymbol{\theta}_c$, and Z is the normalization factor. Intuitively, a configuration of $\boldsymbol{\theta}_c$ with a higher expected constraint value $\mathbb{E}_{\backslash c} T(\boldsymbol{x}^*;\boldsymbol{\theta})$ will receive a higher probability under $q^{(n+1)}(\boldsymbol{\theta}_c)$. Akin to the IPF algorithm (Section 2.1.1), the optimization procedure iterates over all components c of $\boldsymbol{\theta}$.

2.4 Summary

In this section, we have seen that the maximum entropy formalism provides an alternative insight into the classical learning frameworks of MLE, Bayesian inference, and posterior regularization. It provides a general expression of these three paradigms as a constrained optimization problem, with a paradigm-specific loss on the model parameters $\boldsymbol{\theta}$ and an auxiliary distribution q, over a properly designed constraint space $\mathcal{Q}$ where q must reside:

$$\begin{aligned} &\min_{q,\boldsymbol{\theta}} \ \mathcal{L}(q,\boldsymbol{\theta}) \\ &s.t. \ \ q \in \mathcal{Q}. \end{aligned} \tag{2.25}$$

In particular, the use of the auxiliary distribution q converts the originally highly complex problem of directly optimizing $\boldsymbol{\theta}$ against data, to an alternating optimization problem over q and $\boldsymbol{\theta}$, which is algorithmically easier to solve since q often acts as an easy-to-optimize proxy to the target model. The auxiliary q can also be more flexibly updated to absorb influence from data or constraints, offering a teacher-student style iterative mechanism to incrementally update $\boldsymbol{\theta}$ as we will see in the sequel.

By reformulating learning as a constrained optimization problem, the maximum entropy point of view also offers a great source of flexibility for applying many powerful tools for efficient approximation and enhanced learning, such as variational approximation (e.g., by relaxing $\mathcal{Q}$ to be easy-to-inference family of q such as the mean field family (Jordan et al., 1999; Xing et al., 2002)), convex duality (e.g., facilitating dual sparsity of support vectors via the complementary slackness in the KKT conditions), and kernel methods as used in (Taskar et al., 2004; Zhu and Xing, 2009).

It is intriguing that, in the dual point of view on the problem of (supervised) MLE, data instances are encoded as constraints (Eq.2.4), much like the structured constraints in posterior regularization. In the following sections, we present the standardized formalism of machine learning algorithms and show that indeed a myriad types of experiences besides data instances and constraints can all be encoded in the same generic form and be used in learning.

3 The Standard Equation

Generalizing from Eq.(2.21), we present the following general formulation for learning target model via a constrained loss minimization program. We would refer to the formulation as the "standard equation" (SE) because it presents a general space of learning algorithms which encompasses many specific formalisms used in different machine learning paradigms.

Without loss of generality, let $\boldsymbol{t} \in \mathcal{T}$ be the variable of interest, e.g., the input-output pair $\boldsymbol{t} = (\boldsymbol{x}, \boldsymbol{y})$ in a prediction task or the target variable $\boldsymbol{t} = \boldsymbol{x}$ in generative modeling. Let $p_\theta(\boldsymbol{t})$ be the target model to be learned, and $q(\boldsymbol{t})$ be an auxiliary distribution. The SE is written as:

$$\begin{aligned} \min_{q, \boldsymbol{\theta}, \boldsymbol{\xi}} \quad & -\alpha \mathbb{H}\left(q\right) + \beta \mathbb{D}\left(q, p_\theta\right) + U(\boldsymbol{\xi}) \\ s.t. \quad & -\mathbb{E}_q\left[f_k\right] \leq \xi_k, \ \ k = 1, \ldots, K. \end{aligned} \tag{3.1}$$

The SE contains three major terms that constitute a learning formalism: the uncertainty term $\mathbb{H}\left(\cdot\right)$ that controls the compactness of the output model (e.g., as measured by the amount of allowed randomness while trying to fit data); the divergence term $\mathbb{D}\left(\cdot, \cdot\right)$ which measures the distance between the target model to be trained and the auxiliary model that facilitates a teacher-student mechanism as shown below; and a penalty term $U(\boldsymbol{\xi})$ that draws in a set of "experience functions" f_k that represent external experiences of various kinds for training the target model. The hyperparameters $\alpha, \beta \geq 0$ enable trade-offs between these components.

Experience function Perhaps the most powerful in terms of impacting the learning outcome and utility is the experience functions f_k. An experience function $f(\boldsymbol{t}) \in \mathbb{R}$ measures the goodness of a configuration $\boldsymbol{t}$ in light of any given experiences. As discussed in Section 4, all diverse forms of experiences that can be utilized for model training, such as data examples, constraints, logical rules, rewards, and adversarial discriminators, can be encoded as an experience function. The experience function provides a unified language to express all exogenous information about the target model, based on which a standardized optimization program as above can be formulated to identify the desired model. Specifically, the experience function contribute to the optimization objective via the penalty term $U(\boldsymbol{\xi})$ over slack variables $\boldsymbol{\xi} \in \mathbb{R}^K$ applied to the expectation $\mathbb{E}_q\left[f_k\right]$. The effect of maximizing the expectation is such that the auxiliary model q is encouraged to produce samples of high quality in light of the experiences (i.e., samples receiving high scores as evaluated by the experience function).

Assuming a common choice of the penalty $U(\boldsymbol{\xi}) = \sum_k \xi_k$, and, with a slight abuse of notations, $f = \sum_k f_k$, we can re-write Eq.(3.1) in an unconstrained form:

$$\min_{q, \boldsymbol{\theta}} \ -\alpha \mathbb{H}\left(q\right) + \beta \mathbb{D}\left(q, p_\theta\right) - \mathbb{E}_q\left[f\right], \tag{3.2}$$

where the interplay between the exogenous experience, divergence, and the endogenous uncertainty become more explicit.

Teacher-student mechanism The introduction of the auxiliary distribution q relaxes the learning problem of p_θ, originally only over $\boldsymbol{\theta}$, to be now alternating between q and $\boldsymbol{\theta}$. Here q acts as a conduit between the exogenous experience and the target model: it on the one hand subsumes the experience (by maximizing the expected f value), and on the other hand passes it incrementally to the target model (by minimizing the divergence $\mathbb{D}$). The following fixed point iteration between q and $\boldsymbol{\theta}$ illustrates this optimization strategy under the SE. Let us plug into Eq.(3.2) the popular cross entropy (CE) as the divergence measure, i.e., $\mathbb{D}(q, p_\theta) = -\mathbb{E}_q[\log p_\theta]$, and Shannon entropy as the uncertainty measure, i.e, $\mathbb{H}(q) = -\mathbb{E}_q[\log q]$. We have, at iteration n:

$$\begin{aligned} \text{Teacher:} \quad & q^{(n+1)}(\boldsymbol{t}) = \exp\left\{ \frac{\beta \log p_{\theta^{(n)}}(\boldsymbol{t}) + f(\boldsymbol{t})}{\alpha} \right\} \Big/ Z \\ \text{Student:} \quad & \boldsymbol{\theta}^{(n+1)} = \operatorname*{argmax}_{\boldsymbol{\theta}} \mathbb{E}_{q^{(n+1)}(\boldsymbol{t})}\big[\log p_\theta(\boldsymbol{t})\big], \end{aligned} \tag{3.3}$$

where Z is the normalization factor. The first step embodies a "teacher's update" where the teacher q ingests experiences f and builds on current states of the student $p_{\theta^{(n)}}$; the second step is reminiscent of a "student's update" where the student p_θ updates its states by maximizing its alignment (here measured by CE) with the teacher.

Besides, the auxiliary q is an easy-to-manipulate intermediate form in the training that permits rich approximate inference tools for tractable optimization. We have the flexibility of choosing its surrogate functions, ranging from the principled variational approximations for the target distribution in a properly relaxed space (e.g., mean fields) where gaps and bounds can be characterized, to the arbitrary neural network-based "inference networks" that are highly expressive and easy to compute. As can be easily shown (e.g., see Section 4.1.3), popular training heuristics, such as EM, VEM, Wake-Sleep, forward and backward propagation, etc., are all direct instantiations or variants of the above teacher-student mechanism with different choices of the form of q.

More generally, a broad set of sophisticated algorithms, such as the policy gradient for reinforcement learning and the generative adversarial learning, can also be easily derived by plugging specific designs of the experience function f and divergence $\mathbb{D}$. As we will show in the subsequent sections, the standard equation Eq.(3.2), together with the teacher-student mechanism Eq.(3.3), offers a unified and universal paradigm for model training under many scenarios based on many types of experiences, potentiating a turnkey implementation and a more generalizable theoretical characterization.

4 The Experience Function

The experience function $f(\boldsymbol{t})$ in the standard equation can be instantiated to encode vastly distinct types of experiences. Different choices of $f(\boldsymbol{t})$ result in learning algorithms applied to different problems. With particular choices, the standard equation rediscovers a wide array of well-known algorithms. The resulting common treatment of the previously disparate algorithms is appealing as it offers new holistic insights into the commonalities and differences of those algorithms. Table 1 shows examples of extant algorithms that are recovered by the standard equation.

4.1 SE with data instance experiences

We first consider the most common type of experience, namely data instances, which can appear in a wide range of contexts including supervised, self-supervised, unsupervised, actively supervised, and other scenarios with data augmentation and manipulation.

4.1.1 Supervised data instances

Without loss of generality, and for consistency of notations with the rest of the section, we denote the data instance as a pair $\boldsymbol{t} = (\boldsymbol{x}, \boldsymbol{y})$. In the supervised setting, we observe the full data drawn from the data distribution $(\boldsymbol{x}^*, \boldsymbol{y}^*) \sim p_d(\boldsymbol{x}^*, \boldsymbol{y}^*)$. For an arbitrary configuration $(\boldsymbol{x}, \boldsymbol{y})$, its probability $p_d(\boldsymbol{x}, \boldsymbol{y})$ can be seen as measuring the expected similarity between $(\boldsymbol{x}, \boldsymbol{y})$ and true data $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, and be re-written as $p_d(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{p_d(\boldsymbol{x}^*, \boldsymbol{y}^*)}\left[\mathbb{I}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y})\right]$. Here the similarity measure is $\mathbb{I}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y})$, an indicator function that takes the value 1 if $(\boldsymbol{x}, \boldsymbol{y})$ equals $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ and 0 otherwise (we will see other similarity measures shortly). In practice, we are given an empirical distribution $\tilde{p}_d(\boldsymbol{x}, \boldsymbol{y})$ by observing a collection of instances $\mathcal{D}$ on which the expected similarity is evaluated:

$$\mathbb{E}_{(\boldsymbol{x}^*, \boldsymbol{y}^*) \sim \mathcal{D}}\left[\mathbb{I}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y})\right] = \frac{m(\boldsymbol{x}, \boldsymbol{y})}{N}, \tag{4.1}$$

where N is the size of the dataset $\mathcal{D}$, and $m(\boldsymbol{x}, \boldsymbol{y})$ is the number of occurrences of the configuration $(\boldsymbol{x}, \boldsymbol{y})$ in $\mathcal{D}$.

The experience function f accommodates the data instance experience straightforwardly as below:

$$f := f_{\text{data}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) = \log \mathbb{E}_{(\boldsymbol{x}^*, \boldsymbol{y}^*) \sim \mathcal{D}}\left[\mathbb{I}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y})\right]. \tag{4.2}$$

That is, the logarithm of the expected similarity is used as the experience function score, i.e., the more "similar" a configuration $(\boldsymbol{x}, \boldsymbol{y})$ is to the observed data instances, the higher its quality. The logarithm serves to the subsequent derivations more convenient as can be seen below.

With this from of f, we show that the SE derives the conventional supervised MLE algorithm.

Supervised MLE In the SE Eq.(3.2) (with cross entropy and Shannon entropy), we set $\alpha = 1$, and β to a very small positive value ϵ. As a result, the auxiliary distribution $q(\boldsymbol{x}, \boldsymbol{y})$ is determined directly by the full data instances (not the model p_θ). That is, the solution of q in the teacher-step (Eq.3.3) is:

$$q(\boldsymbol{x}, \boldsymbol{y}) = \exp\left\{\frac{\beta \log p_\theta(\boldsymbol{x}, \boldsymbol{y}) + f_{\text{data}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})}{\alpha}\right\} / Z \approx \exp\left\{f_{\text{data}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})\right\} / Z = \tilde{p}_d(\boldsymbol{x}, \boldsymbol{y}), \tag{4.3}$$

which reduces to the empirical distribution. The subsequent student-step that maximizes the log-likelihood of samples from q then leads to the supervised MLE updates w.r.t $\boldsymbol{\theta}$.

4.1.2 Self-supervised data instances

Given an observed data instance $\boldsymbol{t}^* \in \mathcal{D}$ in general, one could potentially derive various supervision signals based on the structures of the data and the target model. In particular, one could apply a "split" function that artificially partitions $\boldsymbol{t}^*$ into two parts $(\boldsymbol{x}^*, \boldsymbol{y}^*) = \textit{split}(\boldsymbol{t}^*)$ in different, sometimes stochastic ways. Then the two parts are treated as the input and output for the properly designed target model $p_\theta(\boldsymbol{x}, \boldsymbol{y})$ for supervised MLE as above, by plugging in the slightly altered experience function:

$$f := f_{\text{data-self}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) = \log \mathbb{E}_{\boldsymbol{t}^* \sim \mathcal{D},\ (\boldsymbol{x}^*, \boldsymbol{y}^*) = \textit{split}(\boldsymbol{t}^*)} \left[\mathbb{I}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y}) \right]. \tag{4.4}$$

A key difference from the above standard supervised learning setting is that now the target variable $\boldsymbol{y}$ is not costly obtained labels or annotations, but rather part of the massively available data instances. The paradigm of treating part of observed instance as the prediction target is called "self-supervised" learning (e.g., Lecun and Misra, 2021) and has achieved great success in language and vision modeling. For example, in language modeling (Devlin et al., 2019; Brown et al., 2020), the instance $\boldsymbol{t}$ is a piece of text, and the "split" function usually selects from $\boldsymbol{t}$ one or few words to be the target $\boldsymbol{y}$ and the remaining words to be $\boldsymbol{x}$.

4.1.3 Unsupervised data instances

In the unsupervised setting, for each instance $\boldsymbol{t} = (\boldsymbol{x}, \boldsymbol{y})$, we only observe the $\boldsymbol{x}$ part. That is, we are given a dataset $\mathcal{D} = \{\boldsymbol{x}^*\}$ without the associated $\boldsymbol{y}^*$. The dataset defines the empirical distribution $\tilde{p}_d(\boldsymbol{x})$. The experience can be encoded in the same form as the supervised data (Eq.4.2) but now with only the information of $\boldsymbol{x}^*$:

$$f := f_{\text{data}}(\boldsymbol{x}; \mathcal{D}) = \log \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}} \left[\mathbb{I}_{\boldsymbol{x}^*}(\boldsymbol{x}) \right]. \tag{4.5}$$

Applying the SE to this setting with proper specifications derives the unsupervised MLE algorithm.

Unsupervised MLE The form of Eq.(3.2) is reminiscent of the variational free energy objective in the standard EM for unsupervised MLE (Eq.2.14). We can indeed get exact correspondence by setting $\alpha = \beta = 1$ and imposing the structure $q(\boldsymbol{x}, \boldsymbol{y}) = \tilde{p}_d(\boldsymbol{x}) q(\boldsymbol{y}|\boldsymbol{x})$. The reason for $\beta = 1$, which differs from the specification $\beta = \epsilon$ in the supervised setting, is that the auxiliary distribution q cannot be determined fully by the unsupervised "incomplete" data experience alone, and instead additionally relies on p_θ through the divergence term. Here q is assumed a specialized structure $q(\boldsymbol{x}, \boldsymbol{y}) = \tilde{p}_d(\boldsymbol{x}) q(\boldsymbol{y}|\boldsymbol{x})$ where $\tilde{p}_d(\boldsymbol{x})$ is fixed and thus not influenced by p_θ. In contrast, if no structure of q is assumed, we could potentially obtain an extended, instance-weighted version of EM where each instance $\boldsymbol{x}^*$ is weighted by the marginal likelihood $p_\theta(\boldsymbol{x}^*)$, in line with the previous weighted EM methods for robust clustering (e.g., Gebru et al., 2016; Yu et al., 2011). We defer detailed derivations to the supplementary materials.

4.1.4 Manipulated data instances

Data manipulation, such as re-weighting data instances or augmenting an existing dataset with new instances, is often a crucial step for efficient learning, such as in low data regime or in presence of low-quality datasets (e.g., imbalanced labels). We show the rich data manipulation schemes can be treated as experiences and be naturally encoded in the experience function (Hu et al., 2019a). This is done by extending the data-instance experience function (Eq.4.2), in particular by enriching the similarity metric in different ways. The discussion here generally applies to data instance $\boldsymbol{t}$ of any structures, e.g., $\boldsymbol{t} = (\boldsymbol{x}, \boldsymbol{y})$ or $\boldsymbol{t} = \boldsymbol{x}$.

Data re-weighting Rather than assuming the same importance of all data instances, we can associate each instance $\boldsymbol{t}^*$ with an importance weight $w(\boldsymbol{t}^*) \in \mathbb{R}$, by scaling the above 0/1 indicator function (e.g., Eq.4.2) with the weight:

$$f_{\text{data-w}}(\boldsymbol{t}; \mathcal{D}) := \log \mathbb{E}_{\boldsymbol{t}^* \sim \mathcal{D}} \left[w(\boldsymbol{t}^*) \cdot \mathbb{I}_{\boldsymbol{t}^*}(\boldsymbol{t}) \right]. \tag{4.6}$$

Plugging $f_{\text{data-w}}$ into the SE (Eq.3.2) with the same other specification of supervised MLE ($\alpha = 1, \beta = \epsilon$), we get the update rule of model parameters $\boldsymbol{\theta}$ in the student-step (Eq.3.3):

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{t}^* \sim \mathcal{D}} \left[w(\boldsymbol{t}^*) \cdot \log p_\theta(\boldsymbol{t}^*) \right], \tag{4.7}$$

which is the familiar weighted supervised MLE. The weights w can be specified *a priori* based on heuristics, e.g., using inverse class frequency. In many cases it is desirable to automatically induce and adapt the weights during the course of model training. In Section 7.2, we discuss how the SE framework can easily enable automated data re-weighting by re-using existing algorithms that were designed to solve other seemingly-unrelated problems.

Data augmentation The indicator function $\mathbb{I}$ as the similarity metric restrictively requires exact match between the true $\boldsymbol{t}^*$ and the configuration $\boldsymbol{t}$. Data augmentation arises as an "relaxation" to the similarity metric. Let $a_{\boldsymbol{t}^*}(\boldsymbol{t}) \geq 0$ be a distribution that assigns non-zero probability to not only the exact $\boldsymbol{t}^*$ but also other configurations $\boldsymbol{t}$ related to $\boldsymbol{t}^*$ in certain ways (e.g., all rotated images $\boldsymbol{t}$ of the observed image $\boldsymbol{t}^*$). Replacing the indicator function metric in Eq.(4.2) with the new $a_{\boldsymbol{t}^*}(\boldsymbol{t}) \geq 0$ yields the experience function for data augmentation:

$$f_{\text{data-aug}}(\boldsymbol{t}; \mathcal{D}) := \log \mathbb{E}_{\boldsymbol{t}^* \sim \mathcal{D}} \left[a_{\boldsymbol{t}^*}(\boldsymbol{t}) \right]. \tag{4.8}$$

The resulting student-step updates of $\boldsymbol{\theta}$, keeping $(\alpha = 1, \beta = \epsilon)$ of supervised MLE, is thus:

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{t}^* \sim \mathcal{D},\ \boldsymbol{t} \sim a_{\boldsymbol{t}^*}(\boldsymbol{t})} \left[\log p_\theta(\boldsymbol{t}) \right]. \tag{4.9}$$

The metric $a_{\boldsymbol{t}^*}(\boldsymbol{t})$ can be defined in various ways, leading to different augmentation strategies. For example, setting $a_{\boldsymbol{t}^*}(\boldsymbol{t}) \propto \exp\{R(\boldsymbol{t}, \boldsymbol{t}^*)\}$, where $R(\boldsymbol{t}, \boldsymbol{t}^*)$ is a task-specific evaluation metric such as BLEU for machine translation, results in the reward-augmented maximum likelihood (RAML) algorithm (Norouzi et al., 2016). Besides the manually designed strategies, we can also specify $a_{\boldsymbol{t}^*}(\boldsymbol{t})$ as a parameterized transformation process and learn any free parameters thereof automatically. Notice the same form of the augmentation experience $f_{\text{data-aug}}$ and the re-weighting experience $f_{\text{data-w}}$, where the similarity metrics both include learnable components (i.e., $a_{\boldsymbol{t}^*}(\boldsymbol{t})$ and $w(\boldsymbol{t}^*)$, respectively). Thus the same solution to automated data re-weighting can also be applied for automated data augmentation, as discussed more in Section 7.2.

4.1.5 Actively supervised data instances

Instead of access to data instances $\boldsymbol{x}^*$ with readily available labels $\boldsymbol{y}^*$, in the active supervision setting, we are presented with a large pool of unlabeled instances $\mathcal{D} = \{\boldsymbol{x}^*\}$ as well as a certain budget for querying an oracle (e.g., human annotators) for labeling a limited set of instances. To minimize the need for labeled instances, we need to strategically select queries from the pool according to an informativeness measure $u(\boldsymbol{x}) \in \mathbb{R}$. For example, $u(\boldsymbol{x})$ can be the predictive uncertainty on the instance $\boldsymbol{x}$, quantified by the Shannon entropy of the predictive distribution or the vote entropy based on a committee of predictors (Dagan and Engelson, 1995).

Mapping the standard equation to this setting, we show the informativeness measure $u(\boldsymbol{x})$ is subsumed as part of the experience. Intuitively, $u(\boldsymbol{x})$ encodes our heuristic belief about sample "informativeness". This heuristic is a form of information we inject into the learning system. Denote the oracle as o from which we can draw a label $\boldsymbol{y}^* \sim o(\boldsymbol{x}^*)$. The active supervision experience function is then defined as:

$$f_{\text{active}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D}) := \log \mathbb{E}_{\boldsymbol{x}^* \sim \mathcal{D}, \boldsymbol{y}^* \sim o(\boldsymbol{x}^*)} \left[\mathbb{I}_{(\boldsymbol{x}^*, \boldsymbol{y}^*)}(\boldsymbol{x}, \boldsymbol{y}) \right] + \lambda \cdot u(\boldsymbol{x}), \tag{4.10}$$

where the first term is essentially the same as the supervised data experience function (Eq.4.2) with the only difference that now the label $\boldsymbol{y}^*$ is from the oracle rather than pre-given in $\mathcal{D}$; $\lambda > 0$ is a trade-off parameter. The formulation of the active supervision is interesting as it is simply a combination of the common supervision experience and the informativeness measure in an additive manner.

We plug f_{active} into the SE and obtain the algorithm to carry out learning. The result turns out to recover classical active learning algorithms.

Active learning Specifically, in Eq.(3.2), setting $f = f_{\text{active}}$, and $(\alpha = 1, \beta = \epsilon)$ as in supervised MLE, the resulting teacher-step in Eq.(3.3) for updating $\boldsymbol{\theta}$ is written as

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{x}^* \sim \tilde{p}_d(\boldsymbol{x}) \cdot \exp\{\lambda u(\boldsymbol{x})\},\ \boldsymbol{y}^* \sim o(\boldsymbol{x}^*)} \left[\log p_\theta(\boldsymbol{x}^*, \boldsymbol{y}^*) \right]. \tag{4.11}$$

If the pool $\mathcal{D}$ is large, the update can be carried out by the following procedure: we first pick a random subset $\mathcal{D}_{\text{sub}}$ from $\mathcal{D}$, and select a sample from $\mathcal{D}_{\text{sub}}$ according to the informativeness distribution proportional to $\exp\{\lambda u(\boldsymbol{x})\}$ over $\mathcal{D}_{\text{sub}}$. The sample is then labeled by the oracle, which is finally used to update the target model. By setting λ to a very large value (i.e., a near-zero "temperature" $1/\lambda$), we tend to select the most informative sample from $\mathcal{D}_{\text{sub}}$. The procedure rediscovers the algorithm proposed in (Ertekin et al., 2007) and more generally the pooling-based active learning algorithms (Settles, 2012).

4.2 SE with knowledge-based experiences

Many aspects of problem structures and human knowledge are difficult if not impossible to be expressed through individual data instances. Examples include the knowledge of expected feature values, maximum margin structures (Section 2.3), logical rules, etc. The knowledge generally imposes constraints which we want the target model to satisfy. The experience function in the standard equation is a natural vehicle for incorporating such knowledge constraints in learning. Given a configuration $\boldsymbol{t}$, the experience function $f(\boldsymbol{t})$ measures the degree to which the configuration satisfies the constraints.

As an example, we consider first-order logic (FOL) rules which provide an expressive declarative language to encode complex symbolic knowledge (Hu et al., 2016). More concretely, let $f_{\text{rule}}(\boldsymbol{t})$ be a FOL rule w.r.t the variables $\boldsymbol{t}$. For flexibility, we use soft logic (Bach et al., 2017) to formulate the rule. Soft logic allows continuous truth values from the interval $[0, 1]$ instead of $\{0, 1\}$, and the Boolean logical operators are redefined as:

$$\begin{aligned} A\&B &= \max\{A+B-1, 0\}, \quad A \vee B = \min\{A+B, 1\} \\ A_1 \wedge \cdots \wedge A_N &= \sum\nolimits_i A_i/N, \quad \neg A = 1 - A. \end{aligned} \tag{4.12}$$

Here $\&$ and $\wedge$ are two different approximations to logical conjunction: $\&$ is useful as a selection operator (e.g., $A\&B = B$ when $A = 1$, and $A\&B = 0$ when $A = 0$), while $\wedge$ is an averaging operator. To give a concrete example, consider the problem of sentiment classification, where given a sentence $\boldsymbol{x}$, we want to predict its sentiment $\boldsymbol{y} \in \{\text{negative } 0, \text{positive } 1\}$. A challenge for a sentiment classifier is to understand the contrastive sense within a sentence and capture the dominant sentiment precisely. For example, if a sentence is of structure “A-but-B” with the connective “but”, the sentiment of the half sentence after “but” dominates. Let $\boldsymbol{x}_B$ be the half sentence after “but” and $\tilde{\boldsymbol{y}}_B \in [0, 1]$ the (soft) sentiment prediction over $\boldsymbol{x}_B$ by the current model, a possible way to express the knowledge as a logical rule $f_{\text{rule}}(\boldsymbol{x}, \boldsymbol{y})$ is:

$$\text{has-‘A-but-B’-structure}(\boldsymbol{x}) \Rightarrow \left(\mathbb{I}(\boldsymbol{y}=1) \Rightarrow \tilde{\boldsymbol{y}}_B \ \& \ \tilde{\boldsymbol{y}}_B \Rightarrow \mathbb{I}(\boldsymbol{y}=1)\right), \tag{4.13}$$

where $\mathbb{I}(\cdot)$ is an indicator function that takes 1 when its argument is true, and 0 otherwise. Given an instantiation (a.k.a. grounding) of $(\boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{y}}_B)$, the truth value of $f_{\text{rule}}(\boldsymbol{x}, \boldsymbol{y})$ can be evaluated by definitions in Eq.(4.12). Intuitively, the $f_{\text{rule}}(\boldsymbol{x}, \boldsymbol{y})$ truth value gets closer to 1 when $\boldsymbol{y}$ and $\tilde{\boldsymbol{y}}_B$ are more consistent.

We then make use of the knowledge-based experiences such as $f_{\text{rule}}(\boldsymbol{t})$ to drive learning. The standard equation rediscovers classical algorithms for learning with symbolic knowledge.

Posterior regularization and extensions By setting $\alpha = \beta = 1$ and f to a constraint function such as f_{rule}, the SE with cross entropy naturally leads to a generalized posterior regularization framework (Hu et al., 2016):

$$\min_{\boldsymbol{\theta}, q} \ -\mathrm{H}\left(q(\boldsymbol{t})\right) - \mathbb{E}_{q(\boldsymbol{t})}\left[\log p_\theta(\boldsymbol{t})\right] - \mathbb{E}\left[f_{\text{rule}}(\boldsymbol{t})\right], \tag{4.14}$$

which extends the conventional Bayesian inference formulation (Section 2.3) by permitting regularization on arbitrary random variables of arbitrary models (e.g., deep neural networks) with complex rule constraints.

The trade-off hyper-parameters can also take other values. For example, by allowing arbitrary $\alpha \in \mathbb{R}$, the objective corresponds to the unified expectation maximization (UEM) algorithm (Samdani et al., 2012) that extends the posterior regularization for added flexibility.

4.3 SE with reward experiences

We now consider a very different learning setting commonly seen in robotic control and other sequential decision making problems. In this setting, experiences are gained by the agent interacting with external environment and collecting feedback in the form of rewards. Formally, we consider a Markov decision process (MDP), where $\boldsymbol{t} = (\boldsymbol{x}, \boldsymbol{y})$ is the state-action pair. More specifically, at time t, the environment is in state $\boldsymbol{x}_t$. The agent draws an action $\boldsymbol{y}_t$ according to the policy $p_\theta(\boldsymbol{y}|\boldsymbol{x})$. The state subsequently transits to $\boldsymbol{x}_{t+1}$ following certain transition dynamics of the environment, and yields a reward $r_t = r(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathbb{R}$. The general goal of the agent is to learn the policy $p_\theta(\boldsymbol{y}|\boldsymbol{x})$ to maximize the reward in the long run. There could be different specifications of the goal. In this section we focus on the one where we want to maximize the expected discounted reward starting from a state drawn from an arbitrary state distribution $p_0(\boldsymbol{x})$, with a discount factor $\gamma \in [0,1]$ applied to future rewards. We discuss other possible goal specifications and their standard equation formulations in the appendix.

A base concept that plays a central role in characterizing the learning in this setting is the action value function, which is the expected discounted future reward of taking action $\boldsymbol{y}$ in state $\boldsymbol{x}$ and continuing with the policy p_θ:

$$Q^\theta(\boldsymbol{x},\boldsymbol{y}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid \boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{y}_0 = \boldsymbol{y}\right], \tag{4.15}$$

where the expectation is taken by following the state dynamics induced by the policy (thus the dependence of Q^θ on policy parameters $\boldsymbol{\theta}$). We next discuss how $Q^\theta(\boldsymbol{x},\boldsymbol{y})$ can be used to specify the experience function in different ways, which in turn derives various known algorithms in reinforcement learning (RL) (Sutton and Barto, 2017). Note that here we are primarily interested in learning the conditional model (policy) $p_\theta(\boldsymbol{y}|\boldsymbol{x})$. Yet we can still define the joint distribution as $p_\theta(\boldsymbol{x},\boldsymbol{y}) = p_\theta(\boldsymbol{y}|\boldsymbol{x})p_0(\boldsymbol{x})$.

Policy gradient The first simple way to use the reward signals as experience is by defining the experience function as the logarithm of the expected future reward:

$$f^\theta_{\text{reward},1}(\boldsymbol{x},\boldsymbol{y}) = \log Q^\theta(\boldsymbol{x},\boldsymbol{y}). \tag{4.16}$$

With $\alpha = \beta = 1$, we arrive at the classical policy gradient algorithm (Sutton et al., 2000). To see this, consider the teacher-student optimization procedure in Eq.(3.3), where the teacher-step yields the q solution:

$$q^{(n)}(\boldsymbol{x},\boldsymbol{y}) = p_{\theta^{(n)}}(\boldsymbol{x},\boldsymbol{y}) Q^{\theta^{(n)}}(\boldsymbol{x},\boldsymbol{y}) \;/\; Z, \tag{4.17}$$

and the student-step updates $\boldsymbol{\theta}$ with the gradient at $\boldsymbol{\theta} = \boldsymbol{\theta}^{(n)}$:

$$\begin{aligned}
&\mathbb{E}_{q^{(n)}(\boldsymbol{x},\boldsymbol{y})}\left[\nabla_\theta \log p_\theta(\boldsymbol{x},\boldsymbol{y})\right] + \mathbb{E}_{q^{(n)}(\boldsymbol{x},\boldsymbol{y})}\left[\nabla_\theta f^\theta_{\text{reward},1}(\boldsymbol{x},\boldsymbol{y})\right] \Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} \\
&= 1/Z \cdot \sum_{\boldsymbol{x}} p_0(\boldsymbol{x}) \nabla_\theta \sum_{\boldsymbol{y}} p_\theta(\boldsymbol{y}|\boldsymbol{x}) Q^\theta(\boldsymbol{x},\boldsymbol{y}) \Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} \\
&= 1/Z \cdot \sum_{\boldsymbol{x}} \mu^\theta(\boldsymbol{x}) \sum_{\boldsymbol{y}} Q^\theta(\boldsymbol{x},\boldsymbol{y}) \nabla_\theta p_\theta(\boldsymbol{y}|\boldsymbol{x}) \Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}}.
\end{aligned} \tag{4.18}$$

Here the first equation is due to the log-derivative trick $g\nabla \log g = \nabla g$; and the second equation is due to the policy gradient theorem (Sutton et al., 2000), where $\mu^\theta(\boldsymbol{x}) = \sum_{t=0}^{\infty} \gamma^t p(\boldsymbol{x}_t = \boldsymbol{x})$ is the unnormalized discounted state visitation measure. The final form is exactly the policy gradient up to a multiplication factor $1/Z$.

Policy gradient with intrinsic reward Rewards provided by the extrinsic environment can be sparse in many real-world sequential decision problems. Learning in such problems is thus difficult due to the lack of supervision signals. A method to alleviate the difficulty is to supplement the extrinsic reward with dense intrinsic reward that is generated by the agent itself (i.e., the agent is intrinsically motivated). The intrinsic reward can be induced in various ways, such as the "curiosity"-based reward that encourages the agent to explore novel or "surprising" states (Schmidhuber, 2010; Houthooft et al., 2016; Pathak et al., 2017), or the "optimal reward" which is designed with the goal of encouraging maximum extrinsic reward at the end (Singh et al., 2010; Zheng et al., 2018). Formally, let $r^{in}_t = r^{in}(\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathbb{R}$ be the intrinsic reward at time t with state $\boldsymbol{x}_t$ and action $\boldsymbol{y}_t$. For example, in (Pathak et al., 2017), r^{in}_t is the prediction error (i.e., the "surprise") of the next state $\boldsymbol{x}_{t+1}$. Let $Q^{in,\theta}(\boldsymbol{x},\boldsymbol{y})$ denote the action-value function for the intrinsic reward, defined in a similar way as the extrinsic $Q^\theta(\boldsymbol{x},\boldsymbol{y})$:

$$Q^{in,\theta}(\boldsymbol{x},\boldsymbol{y}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r^{in}_t \mid \boldsymbol{x}_0 = \boldsymbol{x}, \boldsymbol{y}_0 = \boldsymbol{y}\right]. \tag{4.19}$$

It is straightforward to derive the intrinsically-motivated variant of the policy gradient algorithm (and other RL algorithms discussed below), by replacing the standard extrinsic-only $Q^\theta(\boldsymbol{x},\boldsymbol{y})$ in the experience function Eq.(4.16) with the combined $Q^{\theta}(\boldsymbol{x},\boldsymbol{y}) + Q^{in,\theta}(\boldsymbol{x},\boldsymbol{y})$. Let $f^{\theta}_{\text{reward,ex+in}}(\boldsymbol{x},\boldsymbol{y})$ denote the resulting experience function that incorporates both the extrinsic and the additive intrinsic rewards.

We can notice some sort of symmetry between $f^{\theta}_{\text{reward,ex+in}}(\boldsymbol{x},\boldsymbol{y})$ and the actively supervised data experience f_{active} in Eq.(4.10) which augments the standard supervised data experience with the additive informativeness measure $u(\boldsymbol{x})$. The resemblance could naturally inspire mutual exchange between the research areas of intrinsic reward and active learning, e.g., using the active learning informativeness measure as the intrinsic reward r^{in}, as was studied in earlier work (Schmidhuber, 2010; Li et al., 2011; Pathak et al., 2019).

RL as inference With a slightly different use of the reward plus additional approximations, we can recover the known RL-as-inference approach that has a long history of research (e.g., Dayan and Hinton, 1997; Deisenroth et al., 2013; Rawlik et al., 2013; Levine, 2018; Abdolmaleki et al., 2018). Specifically, now we directly set the experience function to be the reward:

$$f^{\theta}_{\text{reward},2}(\boldsymbol{x},\boldsymbol{y}) = Q^{\theta}(\boldsymbol{x},\boldsymbol{y}), \tag{4.20}$$

and set $\alpha = \beta := \tau > 0$. The configuration corresponds to the approach that casts RL as a probabilistic inference problem. To see this, we introduce an additional binary random variable o, with $p(o = 1|\boldsymbol{x},\boldsymbol{y}) \propto \exp\{Q(\boldsymbol{x},\boldsymbol{y})/\tau\}$. Here $o = 1$ is interpreted as the event that maximum reward is obtained, $p(o = 1|\boldsymbol{x},\boldsymbol{y})$ is seen as the "conditional likelihood", and τ is the temperature. The goal of learning is to maximize the marginal likelihood of optimality: $\log p(o = 1)$, which however is intractable to solve. Much like how the standard equation applied to unsupervised MLE provides a surrogate variational objective for the marginal data likelihood (Section 4.1.3), here the standard equation also derives a variational bound for $\log p(o = 1)$ (up to a constant factor) with the above specification of (f, α, β):

$$\begin{aligned} -\log p(o=1) &= -\log \mathbb{E}_{p_{\theta}(\boldsymbol{x},\boldsymbol{y})}\left[p(o=1|\boldsymbol{x},\boldsymbol{y})\right] \\ &\leq -\tau \mathrm{H}\left(q\right) - \tau \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[\log p_{\theta}(\boldsymbol{x},\boldsymbol{y})\right] - \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[Q^{\theta}(\boldsymbol{x},\boldsymbol{y})\right]. \end{aligned} \tag{4.21}$$

Following the teacher-student procedure in Eq.(3.3), the teacher-step produces the q solution:

$$q^{(n)}(\boldsymbol{x},\boldsymbol{y}) = p_{\theta^{(n)}}(\boldsymbol{x},\boldsymbol{y}) \exp\left\{ Q^{\theta^{(n)}}(\boldsymbol{x},\boldsymbol{y})/\tau \right\} \;/\; Z. \tag{4.22}$$

The subsequent student-step involves approximation by fixing $\boldsymbol{\theta} = \boldsymbol{\theta}^{(n)}$ in $Q^{\theta}(\boldsymbol{x},\boldsymbol{y})$ in the above variational objective, and minimizes only $\mathbb{E}_{q^{(n)}(\boldsymbol{x},\boldsymbol{y})}\left[\log p_{\theta}(\boldsymbol{x},\boldsymbol{y})\right]$ w.r.t $\boldsymbol{\theta}$.

4.4 SE with advanced experiences

Besides the regular types of experiences discussed above, the experience function f can also accommodate more advanced forms of experiences, such as trained models of related tasks, and experiences involving complex interactions with the target model through co-training or adversarial dynamics.

As a simple example of using trained models of related tasks as the experience, consider the problem of generating a sentence $\boldsymbol{y}$ of given sentiment $\boldsymbol{x}$ (positive or negative). We can obtain experiences from the related sentiment classification task, by using a trained sentiment classifier as the experience function $f(\boldsymbol{x},\boldsymbol{y})$ which measures the probability of a sentence $\boldsymbol{y}$ being of sentiment $\boldsymbol{x}$. We discuss more in Section 7.1, where the SE framework enables to combine multiple trained models from different related tasks to train the model for the target task.

Some sophisticated experiences may even not have an analytic form but instead is defined in a variational way, i.e.,

$$f := \operatorname*{argmax}_{f} \mathcal{J}(f), \tag{4.23}$$

with specific optimization objective $\mathcal{J}$. A concrete example is the adversarial experience emergingly used in many generation and representation learning problems. Specifically, the experiences discussed above are mostly defined *a priori* and encoded as a fixed experience function $f(\boldsymbol{t})$. For example, the data instance experience $f_{\text{data}}(\boldsymbol{t};\mathcal{D})$ measures the similarity or closeness between a configuration $\boldsymbol{t}$ with the true data $\mathcal{D}$ based on data instance matching (Eq.4.2). Such manually-defined measures could be subjective, sub-optimal, or demanding expertise or heavy engineering to be properly defined. An alternative way that sidesteps the drawbacks is to automatically induce a measure $f_\phi(\boldsymbol{t})$, where $\boldsymbol{\phi}$ denote any free parameters associated with the experiences and to be learned. For example, one can measure the closeness of a configuration $\boldsymbol{t}$ to the dataset $\mathcal{D}$ based on a discriminator (or critic) that evaluates how easily $\boldsymbol{t}$ can be differentiated from the instances in $\mathcal{D}$. The similar idea of discriminator-based closeness measure was explored in the likelihood-free inference literature (Gutmann et al., 2014). The critic as the experience can be learned or adapted during the course of model training. We show in the next section that the critic-based experience, in combination of certain choices of the divergence measure $\mathbb{D}$, produces the popular generative adversarial learning (GANs, Goodfellow et al., 2014).

Similarly, as briefly mentioned earlier, many of the above conventional experiences can also benefit from the idea of introducing adaptive or learnable components, e.g., data instances with automatically induced data weights or learned augmentation policies. We discuss in Section 7.2 how the unified SE framework can naturally derive efficient solutions for joint model and experience training.

5 The Divergence Measure

We now turn to the divergence term $\mathbb{D}(q, p_\theta)$ in the standard equation. The discussion in the prior section has focused on the specific cases of $\mathbb{D}$ being the cross entropy. Yet there is a rather rich set of choices for the divergence measure, such as f-divergence (e.g., KL divergence, Jensen-Shannon divergence), optimal transport distance (e.g., Wasserstein distance), etc.

To see a concrete setting of how the divergence measure may influence the learning, consider the experience to be data instances with the data distribution $p_d(\boldsymbol{t})$, and, following the configurations of supervised MLE (Section 4.1.1), set $f = f_{\text{data}}$, $\alpha = 1$ and $\beta = \epsilon$. As a result, the solution of q in the standard equation Eq.(3.2) reduces to the data distribution $q(\boldsymbol{t}) = p_d(\boldsymbol{t})$ (assuming the uncertainty measure $\mathbb{H}$ to be the Shannon entropy). The learning of model thus reduces to minimizing the divergence between the model and data distributions:

$$\min_{\boldsymbol{\theta}} \mathbb{D}(p_d, p_\theta), \tag{5.1}$$

which is a commonly seen objective shared by many ML algorithms depending on how the divergence measure is specialized. Thus, in this concrete setting the divergence measure directly determines the learning problem. Below we will see richer influences of $\mathbb{D}$ in other settings in combination with other standard equation components.

By considering certain choices for the divergence measure, we open up the door to recover and generalize some core techniques that are widely practiced in the area of generative adversarial learning. The standard equation offers two different ways of deriving the generative adversarial learning algorithms, where the key concept, discriminator, arises either as an approximation to the optimization procedure, or as part of the experience in the learning objective.

5.1 Generative adversarial learning

The functional descent view Given a specialized divergence $\mathbb{D}$, the objective Eq.(5.1) can be optimized with different solvers, among which probability functional descent (PFD) (Chu et al., 2019) offers an elegant way to derive the optimization which recovers various existing algorithms in GANs (Goodfellow et al., 2014). We give a brief overview here and discuss more details of the optimization techniques in the next section.

Specifically, given p_d, $J(p) := \mathbb{D}(p_d, p)$ is a functional on the distribution space $\mathcal{P}(\mathcal{T})$. The Gâteaux derivative of J at p, if exists, is defined as (Fernholz, 2012):

$$J'_p(h - p) = \lim_{\epsilon \to 0^+} \frac{J(p + \epsilon(h - p)) - J(p)}{\epsilon} \tag{5.2}$$

for any given $h \in \mathcal{P}(\mathcal{T})$. Intuitively, $J'_p(h - p)$ describes the change of the $J(p)$ value with respect to an infinitesimal change in p in the direction of $(h - p)$. The Gâteaux derivative $J'_p(h - p)$ can alternatively be computed with the influence function of J at p, denoted as $\psi_p : \mathcal{T} \to \mathbb{R}$, through:

$$J'_p(h - p) = \int_{\boldsymbol{t}} \psi_p(\boldsymbol{t})(h - p) d\boldsymbol{t} = \mathbb{E}_h \left[\psi_p(\boldsymbol{t})\right] - \mathbb{E}_p \left[\psi_p(\boldsymbol{t})\right]. \tag{5.3}$$

The above notions allow us to define gradient descent applied to the functional J. Concretely, we can define a linear approximation to $J(p)$ around a given p_0:

$$\begin{aligned} J(p) &\approx J(p_0) + J'_{p_0}(p - p_0) \\ &= J(p_0) + \mathbb{E}_p \left[\psi_{p_0}(\boldsymbol{t})\right] - \mathbb{E}_{p_0} \left[\psi_{p_0}(\boldsymbol{t})\right] \\ &= \mathbb{E}_p \left[\psi_{p_0}(\boldsymbol{t})\right] + const. \end{aligned} \tag{5.4}$$

Thus, $J(p)$ can approximately be minimized with an iterative descent procedure: at each iteration n, we perform a descent step that decreases $\mathbb{E}_p \left[\psi_{p^{(n)}}(\boldsymbol{t})\right]$ w.r.t p, yielding $p^{(n+1)}$ for the next iteration. Note that, in practice where the model distribution of interest p is often parameterized as p_θ, chain rule can be applied assuming mild regularity conditions to compute the gradient $\nabla_\theta J(p_\theta) \approx \nabla_\theta \mathbb{E}_{p_\theta}[\psi_{p_{\theta(n)}}(\boldsymbol{t})]$.

Once the functional gradient is defined as above, the remaining problem of the optimization is then about how to obtain the influence function ψ_p given the functional $J(p)$. In some cases the influence function as defined in Eq.(5.3) is not directly tractable and approximations are needed. Chu et al. (2019) developed a variational approximation method applied when J is convex. Concretely, with the convex conjugate of J defined as $J^*(\varphi) = \sup_h \mathbb{E}_h \left[\varphi(\boldsymbol{t})\right] - J(h)$, it can be shown under mild conditions that the influence function for J at p is:

$$\psi_p = \mathrm{argmax}_{\varphi \in \mathcal{C}(\mathcal{T})} \mathbb{E}_p \left[\varphi(\boldsymbol{t})\right] - J^*(\varphi), \tag{5.5}$$

where $\mathcal{C}(\mathcal{T})$ is the space of continuous functions $\mathcal{T} \to \mathbb{R}$. We thus can approximate the influence function by parameterizing it as a neural network and training the network to maximize the objective $\mathbb{E}_p \left[\varphi(\boldsymbol{t})\right] - J^*(\varphi)$. Plugging the approximation of influence function into the above functional descent procedure leads to the full PFD optimization:

$$\inf_p \sup_\varphi \mathbb{E}_p \left[\varphi(\boldsymbol{t})\right] - J^*(\varphi). \tag{5.6}$$

The saddle-point problem is reminiscent of the generative adversarial learning. In fact, as shown in (Chu et al., 2019), when $J(p) = \mathbb{D}(p_d, p)$ is the Jensen-Shannon divergence and φ is parameterized as $\varphi_\phi = \frac{1}{2}\log(1 - C_\phi) - \frac{1}{2}\log 2$ where C_ϕ is a binary classifier (discriminator), then Eq.(5.6) recovers the original GAN algorithm (Goodfellow et al., 2014):

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \frac{1}{2}\mathbb{E}_{p_d} \left[\log C_\phi(\boldsymbol{t})\right] - \frac{1}{2}\mathbb{E}_{p_\theta} \left[\log(1 - C_\phi(\boldsymbol{t}))\right]. \tag{5.7}$$

On the other hand, consider the case of Wasserstein GAN algorithm (Arjovsky et al., 2017) where $\mathbb{D}(p_d, p)$ is set to the first-order Wasserstein distance $W_1(p, p_d)$:

$$\inf_p J(p) := \inf_p W_1(p, p_d) = \inf_p \sup_{\|\varphi\|_L \le 1} \mathbb{E}_p \left[\varphi(\boldsymbol{t})\right] - \mathbb{E}_{p_d} \left[\varphi(\boldsymbol{t})\right], \tag{5.8}$$

where the last equation is due to the Kantorovich duality (Santambrogio, 2015, Section 1.2) and $\|\varphi\|_L \le 1$ is the constraint of φ being a 1-Lipschitz function. We can also interpret Eq.(5.8) using the PFD procedure. Specifically, due to the Fenchel–Moreau theorem saying that $J(p) = \sup_\varphi \mathbb{E}_p \left[\varphi(\boldsymbol{t})\right] - J^*(\varphi)$, we have $J^*(\varphi) = \mathbb{E}_{p_d} \left[\varphi(\boldsymbol{t})\right] + \mathbb{I}_\infty(\|\varphi\|_L \le 1)$, where $\mathbb{I}_\infty(A)$ is an indicator function that equals 1 if A is true and ∞ otherwise. Plugging into Eq.(5.5) gives the expression of the influence function, which corresponds to the Kantorovich potential for the transport from p to p_d (Santambrogio, 2015, Proposition 7.17).

The variational experience view The functional descent view of generative adversarial learning is based on the treatment that the experience is the given data instances, so the various GAN algorithms are due to the different choices of the divergence measure. The standard equation also allows an alternative viewpoint of the learning paradigm, that gives more flexibility in not only choosing the divergence measure but also the experience function, leading to a richer set of GAN extensions.

In the alternative viewpoint, we consider experience that is variationally defined as discussed in Section 4.4. That is, the experience function f, as a measure of the goodness of a sample $\boldsymbol{t}$, is not specified *a priori* but rather defined through an optimization problem. A concrete example is to define f as a binary classifier f_ϕ with sigmoid activation and parameters $\boldsymbol{\phi}$, where the value $f_\phi(\boldsymbol{t})$ measures the *log* probability of the sample $\boldsymbol{t}$ being a real instance (as opposed to a model generation). Thus the higher $f_\phi(\boldsymbol{t})$ value, the higher quality of sample $\boldsymbol{t}$. The parameters $\boldsymbol{\phi}$ of the experience function need to be learned. We can do so by augmenting the standard equation (Eq.3.2) with added optimization of $\boldsymbol{\phi}$ in various ways. The following equation gives one of the approaches:

$$\min_{q,\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \; -\alpha \mathbb{H}(q) + \beta \mathbb{D}(q, p_\theta) - \mathbb{E}_q[f_\phi] + \mathbb{E}_{p_d}[f_\phi], \tag{5.9}$$

where, besides the optimization of q and $\boldsymbol{\theta}$, we additionally maximize over $\boldsymbol{\phi}$ with the extra term $\mathbb{E}_{p_d}[f_\phi]$ to form the classification problem $\max_{\boldsymbol{\phi}} -\mathbb{E}_q[f_\phi] + \mathbb{E}_{p_d}[f_\phi]$. Further assuming a particular configuration of the tradeoff hyper-parameters $\alpha = 0$ and $\beta = 1$, the resulting objective

$$\min_{q,\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \; \mathbb{D}(q, p_\theta) - \mathbb{E}_q[f_\phi] + \mathbb{E}_{p_d}[f_\phi], \tag{5.10}$$

turns out to relate closely to generative adversarial learning.

In particular, with proofs adapted from (Farnia and Tse, 2018), Eq.(5.10) recovers the vanilla GAN algorithm when $\mathbb{D}$ is the Jensen-Shannon divergence and assuming the space of f_ϕ, denoted as $\mathcal{F}$, is convex. More specifically, if we denote the probability $C_\phi(\boldsymbol{t}) = \exp f_\phi(\boldsymbol{t})$, then the equation reduces to the familiar GAN objective in Eq.(5.7). The results can be extended to the more general case of f-GAN (Nowozin et al., 2016): if we set $\mathbb{D}$ to an f-divergence and do not restrict the form (e.g., classifier) of the experience function f_ϕ, then with mild conditions, the equation recovers the f-GAN algorithm. Now consider $\mathbb{D}$ as the first-order Wasserstein distance and suppose the f_ϕ-space $\mathcal{F}$ is a convex subset of 1-Lipschitz functions. It can be shown that Eq.(5.10) reduces to the Wasserstein GAN algorithm as shown in Eq.(5.8) where φ now corresponds to f_ϕ. Note that for the above configurations, if f_ϕ is parameterized as a neural network with a fixed architecture (e.g., ConvNet), its space $\mathcal{F}$ is not necessarily convex (i.e., a linear combination of two neural networks in $\mathcal{F}$ is not necessarily in $\mathcal{F}$). In such cases we formulate the optimization of the experience function over $\mathrm{conv}(\mathcal{F})$, the convex hull of $\mathcal{F}$ containing any convex combination of neural network functions in $\mathcal{F}$ (Farnia and Tse, 2018), and see the various GAN algorithms as approximations by considering only the subset $\mathcal{F} \subseteq \mathrm{conv}(\mathcal{F})$.

Besides the above examples of divergence $\mathbb{D}$ that each leads to a different GAN algorithm, we can consider even more options, such as the hybrid f-divergence and Wasserstein distance studied in (Farnia and Tse, 2018). Of particular interest is to set $\mathbb{D}$ to the KL divergence $\mathbb{D}(q, p_\theta) = \mathrm{KL}(q \| p_\theta)$, motivated by the simplicity in the sense that the auxiliary distribution q has a closed-form solution:[1] at each iteration n,

$$q^{(n+1)}(\boldsymbol{t}) = p_{\theta^{(n)}}(\boldsymbol{t}) \exp\left\{ f_{\phi^{(n)}}(\boldsymbol{t}) \right\} \; / \; Z, \tag{5.11}$$

where Z is the normalization factor. As shown in Wu et al. (2020), the particular form of solution results in a new variant of GANs that enables more stable optimization of the experience function (discriminator) f_ϕ. Concretely, the discriminator f_ϕ can now be optimized with importance reweighting:

$$\begin{aligned} &\max_{\boldsymbol{\phi}} - \mathbb{E}_{\boldsymbol{t}\sim q^{(n+1)}}[f_\phi(\boldsymbol{t})] + \mathbb{E}_{\boldsymbol{t}\sim p_d}[f_\phi(\boldsymbol{t})] \\ &\quad = -\frac{1}{Z}\mathbb{E}_{\boldsymbol{t}\sim p_\theta^{(n)}}\Big[\exp\{f_{\phi^{(n)}}(\boldsymbol{t})\} \cdot f_\phi(\boldsymbol{t})\Big] + \mathbb{E}_{\boldsymbol{t}\sim p_d}[f_\phi(\boldsymbol{t})], \end{aligned} \tag{5.12}$$

where importance sampling is used to estimate the expectation under $q^{(n+1)}$, using the generator $p_\theta^{(n)}$ as the proposal distribution. Compared to the vanilla and Wasserstein GANs above, the fake samples from the generator are now weighted by the exponentiated discriminator score $\exp\{f_{\phi^{(n)}}(\boldsymbol{t})\}$ when used to update the discriminator. Intuitively, the mechanism assigns higher weights to samples that can fool the discriminator better, while low-quality samples are downplayed to avoid destructing the discriminator performance during training.

6 Optimization Algorithms

Thus far, we have seen the standard equation of objective function for learning with different experiences. The standard objective presents an optimization problem on which we apply an optimization solver to find the model solution. This section is devoted to discussion of various solving algorithms.

[1] We can alternatively derive the objective from Eq.(5.9), by setting $\alpha = \beta = 1, \mathbb{D}$ to the cross entropy, and $\mathbb{H}$ to the Shannon entropy. Note that $\mathrm{KL}(q\|p_\theta) = -\mathrm{H}(q) - \mathbb{E}_q[\log p_\theta]$. Thus the solution of q can be derived as in Eq.(3.3).

Experience type	Experience function f	Divergence $\mathbb{D}$	α	β	Algorithm
Data instances	$f_{\text{data}}(\boldsymbol{x}; \mathcal{D})$	CE	1	1	Unsupervised MLE
	$f_{\text{data}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})$	CE	1	ϵ	Supervised MLE
	$f_{\text{data-w}}(\boldsymbol{t}; \mathcal{D})$	CE	1	ϵ	Data re-weighting
	$f_{\text{data-aug}}(\boldsymbol{t}; \mathcal{D})$	CE	1	ϵ	Data Augmentation
	$f_{\text{active}}(\boldsymbol{x}, \boldsymbol{y}; \mathcal{D})$	CE	1	ϵ	Active Learning (Ertekin et al., 2007)
Knowledge	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	CE	1	1	Posterior Regularization (Ganchev et al., 2010)
	$f_{rule}(\boldsymbol{x}, \boldsymbol{y})$	CE	$\mathbb{R}$	1	Unified EM (Samdani et al., 2012)
Reward	$\log Q^{\theta}(\boldsymbol{x}, \boldsymbol{y})$	CE	1	1	Policy Gradient
	$\log Q^{\theta}(\boldsymbol{x}, \boldsymbol{y}) + Q^{in,\theta}(\boldsymbol{x}, \boldsymbol{y})$	CE	1	1	+ Intrinsic Reward
	$Q^{\theta}(\boldsymbol{x}, \boldsymbol{y})$	CE	$\tau > 0$	$\tau > 0$	RL as Inference
Other advanced	binary classifier	JSD	0	1	Vanilla GAN (Goodfellow et al., 2014)
	discriminator	f-divg.	0	1	f-GAN (Nowozin et al., 2016)
	1-Lipschitz discriminator	W_1 dist.	0	1	WGAN (Arjovsky et al., 2017)
	1-Lipschitz discriminator	KL	0	1	PPO-GAN (Wu et al., 2020)

Table 1: Example configurations of components in the standard equation (Eq.3.2) which recover existing algorithms. Here, "CE" means Cross Entropy; "JSD" is the Jensen-Shannon divergence; "f-divg" is the f-divergence; "W_1 dist." is the first-order Wasserstein distance; and "KL" is the KL divergence. Refer to Sections 4 and 5 for more details.

We have seen examples of the optimization procedure, such as that of Eq.(3.3) based on a teacher-student mechanism. Like the standardized formulation of objective function, we would like to quest for a standardized solving algorithm that is generally applicable to optimizing the objective under vastly different specifications. It seems still unclear whether such a general solver exists or what it looks like, though some techniques may hold the promise to generalize to broad settings.

For a simple objective such as that of the vanilla supervised MLE (Eq.2.1) with a tractable model, stochastic gradient descent can be used to optimize the model parameters $\boldsymbol{\theta}$ straightforwardly. With a more complex model, such as the undirected graphical model with an intractable normalization factor Z (Eq.2.8), or a more complex objective as the SE, more sophisticated solving algorithms are required. Alternating projection (Csiszár, 1975; Bauschke and Borwein, 1996) provides a general class of solving algorithms from a geometry point of view, subsuming as special cases many of the solving algorithms discussed above, ranging from IPF (Section 2.1.1), EM (Section 2.1.2), to the teacher-student algorithm (Section 3). At a high level, the algorithms consider the optimization problem as to find a common point of a collection of sets, and achieve it by alternatingly projecting between those sets. For example, each update of IPF in Eq.(2.12) is an I-projection of $p^{(n)}(\boldsymbol{x})$ onto the set of distributions that satisfy the marginal constraint $p(\boldsymbol{x}_c) = \tilde{p}_d(\boldsymbol{x}_c)$, which is written as:

$$\begin{aligned} &\min_{p} \ \text{KL}\left(p(\boldsymbol{x}) \| p^{(n)}(\boldsymbol{x})\right) \\ &s.t. \ \ p(\boldsymbol{x}_c) = \tilde{p}_d(\boldsymbol{x}_c). \end{aligned} \tag{6.1}$$

Similarly, the EM algorithm entails alternating projection, where the E-step (Eq.2.15) projects the current model distribution $p_{\theta^{(n)}}(\boldsymbol{x}, \boldsymbol{y})$ onto the set of distributions whose marginal over $\boldsymbol{x}$ equals the empirical distribution, i.e., $q^{(n+1)} = \text{argmin}_q \text{KL}\left(q(\boldsymbol{y}|\boldsymbol{x})\tilde{p}_d(\boldsymbol{x}) \| p_{\theta^{(n)}}(\boldsymbol{x}, \boldsymbol{y})\right)$, then the subsequent M-step (Eq.2.16) projects $q^{(n+1)}$ onto the set of possible model distributions through $\min_\theta \text{KL}\left(q^{(n+1)}(\boldsymbol{y}|\boldsymbol{x})\tilde{p}_d(\boldsymbol{x}) \| p_\theta(\boldsymbol{x}, \boldsymbol{y})\right)$.

In general, the SE Eq.(3.2) with both the model parameters $\boldsymbol{\theta}$ and the auxiliary distribution q to be learned can naturally be optimized with an alternating-projection style procedure. The teacher-student algorithm (Eq.3.3) is one of the examples, where the teacher $q^{(n+1)}$ is the projection of the student $p_{\theta^{(n)}}$ onto the set defined by the experience, and the student $p_{\theta^{(n+1)}}$ is the projection of the teacher $q^{(n+1)}$ onto the set of model distributions. However, the algorithm applies only when cross entropy is used as the divergence measure. With other choices of more complex divergence, we may not have the simple closed-form solution for q at teacher steps, making the projection difficult or infeasible. The probabilistic functional descent (PFD) described in Section 5.1, with approximations to the influence function using convex duality, seems to offer a possible way of solving for q for certain divergences, such as Jensen-Shannon divergence and Wasserstein distance. It presents a promising venue for future research to develop more generic solvers for the broad learning problems characterized by SE.

7 Learning with All Experiences

The preceding sections have presented a standardized formalism of machine learning, on the basis of the standard equation of objective function, that provides a succinct, structured formulation of a broad design space of learning algorithms, and subsumes a wide range of known algorithms in a unified manner. The simplicity, modularity and generality of the framework is particularly appealing not only from the theoretical perspective, but also because it offers guiding principles for designing algorithmic solutions to challenging problems in a mechanic way. In this section, we discuss the use of standard equation to drive systematic design of new learning methods, which in turn yield various algorithmic solutions to problems in different application domains.

7.1 Combining Rich Experiences

As one of the original motivations for the standardization, the framework allows us to combine together all different experiences to learn models of interest. Learning with multiple types of experiences is a necessary capability of AI agents to be deployed in a real dynamic environment and to improve from heterogeneous signals in different forms, at varying granularities, from diverse sources, and with different intents (e.g., adversaries). Such versatile learning capability is best exemplified by human learning, which has the hallmark of making use of any available sources of information. Take the example of learning a language. Humans can benefit from observing examples through reading and hearing, studying abstract definitions and grammar, making mistakes and getting correction from teacher, interacting with others and observing implicit feedback, etc. Knowledge of prior language can also accelerate the acquisition of new one. To build a similar panoramic learning AI agent, having a standardized learning formalism is perhaps an indispensable step.

The standard equation we have presented is naturally designed to fit the needs. In particular, the experience function provides a straightforward vehicle for experience combination. For example, the simplest approach is perhaps to make a weighted composition of multiple individual experience functions:

$$f(\boldsymbol{t}) = \sum\nolimits_i \lambda_i f_i(\boldsymbol{t}), \tag{7.1}$$

where each f_i characterizes a specific source of experiences, and $\lambda_i > 0$ is the respective weight. One can readily plug in arbitrary available experiences, such as data, logical rules, constraints, rewards, and auxiliary models, as components of the experience function.

With the SE, designing a solution to a problem thus boils down to choosing and formulating *what* experiences to use depending on the problem structure and available resources, without worrying too much about *how* to use the experiences. This provides a potentially new modularized and repeatable way of producing ML solutions to different problems, as compared to the previous practice of designing bespoke algorithm for each individual problem. Section 4 has discussed possible formulations of the diverse types of experiences as an experience function to be plugged into Eq.(7.1). It is still an open question how even more types of experiences, such as massive knowledge graphs (Tan et al., 2020; Bach et al., 2019), can efficiently be formulated as an experience function that assesses the "goodness" of a configuration $\boldsymbol{t}$. The discussion in the next section offers new opportunities that allow users to not have to manually specify every details of the experience function, but instead only to specify parts of it the users are certain of, and leave the remaining parts plus the weights λ_i (Eq.7.1) automatically learned together with the target model.

Case study: text attribute transfer As a case study of learning from rich experiences, consider the problem of text attribute transfer where we want to rewrite a given piece of text to possess a desired attribute (Hu et al., 2017; Shen et al., 2017) . Taking the sentiment attribute for example, given a sentence $\boldsymbol{x}$ (e.g., a customer's review "the manager is a horrible person") and a target sentiment a (e.g., positive), the goal of the problem is to generate a new sentence $\boldsymbol{y}$ that (1) possesses the target sentiment, (2) preserves all other characteristics of the original sentence, and (3) is fluent (e.g., "the manager is a perfect person"). To learn an attribute transfer model $p_\theta(\boldsymbol{y}|\boldsymbol{x}, a)$, a key challenge of the problem is the lack of direct supervision data (i.e., pairs of sentences that are exact the same except for sentiment), making it necessary to use other forms of experiences. Here we briefly describe a solution originally presented in (Hu et al., 2017; Yang et al., 2018), highlighting how the solution can be built mechanically, by formulating relevant experiences based on the problem definition and then plugging them into the SE.

We can identify three types of experiences, corresponding to the above three desiderata, respectively. First, the model needs to learn the concept of "sentiment" to be able to modify the attribute of text. A natural form of experience that has encoded the concept is a pretrained sentiment classifier (SC). The first experience function can then be defined as $f_1(\boldsymbol{x}, a, \boldsymbol{y}) = \text{SC}(a, \boldsymbol{y})$, which evaluates the likelihood of the transferred sentence $\boldsymbol{y}$ possessing the sentiment a. Thus, the higher value the $\boldsymbol{y}$ achieves, the higher-quality it is considered in light of the experience. The second desideratum requires the model to reconstruct as much of the input text $\boldsymbol{x}$ as possible. We combine the second experience $f_{\text{data}}(\boldsymbol{x}, a, \boldsymbol{y}|\mathcal{D})$ (Eq.4.2) defined by a set of simple reconstruction data instances $\mathcal{D} = \{(\boldsymbol{x}^*, a^* = a_{\boldsymbol{x}^*}, \boldsymbol{y}^* = \boldsymbol{x}^*)\}$, where the target sentiment a^* is set to the sentiment of the original sentence $\boldsymbol{x}^*$, and by the problem definition the ground-truth output $\boldsymbol{y}^*$ is exact the same as the input $\boldsymbol{x}^*$. Finally, for the requirement of fluent generation, we can again naturally use an auxiliary model as the experience, namely a pretrained language model (LM) $f_3(\boldsymbol{x}, a, \boldsymbol{y}) = \text{LM}(\boldsymbol{y})$ that estimates the likelihood of a sentence $\boldsymbol{y}$ under the natural language distribution.

7.2 Repurposing Learning Algorithms for New Problems

The standardized formalism sheds new lights on fundamental relationships between a number of learning problems in different research areas, showing that they are essentially the same under the SE perspective. This opens up a wide range of opportunities for generalizing existing algorithms, which were originally designed for specialized problems, to a much broader set of new problems. It is also made easy to exchange between the diverse research areas in aspects of modeling, theoretical understanding, and approximation and optimization tools. For example, an earlier successful solution to challenges in one area can now be readily applied to address challenges in another. Similarly, a future progress made in one problem could immediately unlock progresses in many others.

Case study: learning with imperfect experiences To illustrate, let us consider a set of concrete problems concerning with distinct types of experiences, which were often studied by researchers in different areas: **(1)** The first problem is to integrate structured knowledge constraints in model training, where some components of the constraints, as well as the constraint weights, cannot be specified *a priori* and are to be induced automatically; **(2)** The second problem concerns supervised learning, where one has access to only a small set of data instances with imbalanced labels, and we want to automate the data manipulation (e.g., augmentation and re-weighting) to maximize the training performance. **(3)** The last problem is to stabilize the notoriously difficult training of generative adversarial networks (GANs) for a wide range of image and text generation tasks.

The three problems, though seemingly unrelated at first sight, can all be reduced to the same underlying problem in the unified SE view, namely, learning with imperfect experience f (e.g., underspecified knowledge constraints, small imbalanced data, and unstable discriminator). We want to automatically adapt/improve the imperfect experience in order to better supervise the target model training. This is formulated as an extension to Eq.(3.2), where now the experience function is $f_\phi(\boldsymbol{t})$ associated with learnable parameters $\boldsymbol{\phi}$ (see also Section 4.4). For example, in problem (1), $f_\phi(\boldsymbol{t}) = \sum_i \lambda^i_\phi f^i_{\text{rule},\phi}(\boldsymbol{t})$ where any learnable components in each knowledge constraint $f^i_{\text{rule},\phi}$ (Section 4.2) and the weights λ^i_ϕ constitute the $\boldsymbol{\phi}$ to be learned (Hu et al., 2018). In problem (2), $f_\phi(\boldsymbol{t})$ is instantiated as $f_{\text{data-w},\phi}(\boldsymbol{t}; \mathcal{D})$ (Eq.4.6) with learnable data weights $w(\boldsymbol{t}^*) \in \boldsymbol{\phi}$, or as $f_{\text{data-aug},\phi}(\boldsymbol{t}; \mathcal{D})$ (Eq.4.8) with the metric for augmentation $a_{\boldsymbol{t}^*}(\boldsymbol{t}) \in \boldsymbol{\phi}$ to be learned (Hu et al., 2019a). In problem (3), we have discussed the training of $f_\phi(\boldsymbol{t})$ as the GAN discriminator in Section 5.1, but we want to improve the training stability (Wu et al., 2020). Thus, one solution for efficient updates of the general experience function f_ϕ would address all the three problems together.

To seek for solutions, we again take advantage of the unified SE perspective that enables us to reuse existing successful techniques instead of having to invent new ones. In particular, the connection of the experience function f with the reward in Section 4.3 naturally inspires us to repurpose known techniques from the fertile reinforcement learning (RL) literature, especially those of learning reward functions such as inverse RL (Ziebart et al., 2008) or learning implicit reward (Zheng et al., 2018), for learning the experience function f_ϕ in our problems. For instance, following (Ziebart et al., 2008), one can augment the SE training Eq.(3.3) with an additional step for updating $\boldsymbol{\phi}$ (Hu et al., 2018) through $\min_{\boldsymbol{\phi}} -\mathbb{E}_{\boldsymbol{t}^* \sim \tilde{p}_{\text{data}}} [\log q(\boldsymbol{t}^*)]$, where q taking the form in Eq.(3.3) now depends on $\boldsymbol{\phi}$. The resulting procedure induces an importance re-weighting scheme that is shown to stabilize the discriminator training in GANs (Wu et al., 2020).

8 Related Work

It has been a constant aspiration to search for basic principles that unify the different paradigms in machine learning (Langley, 1989; Bishop, 2013; Domingos, 2015; Gori, 2017; Hu et al., 2019b). Extensive efforts have been made to build unifying views of methods on particular fronts. For example, Roweis and Ghahramani (1999) unified various unsupervised learning algorithms with a linear Gaussian model; Wainwright and Jordan (2005) presented the variational method for inference in general exponential-family graphical models; Knoblauch et al. (2019) developed a generalized form of variational Bayesian inference by allowing losses and divergences beyond the standard likelihood and KL divergence, which subsumes existing variants for Bayesian posterior approximation; Altun and Smola (2006) showed the duality between regularized divergence (e.g., Bregman and f-divergence) minimization and statistical inference (e.g., MAP estimation); Arora et al. (2012) presented a common formulation of different multiplicative weights update methods; Mohamed and Lakshminarayanan (2016) connected generative adversarial learning with a rich set of other statistical learning principles; and so forth (Chu et al., 2019; Wu et al., 2019; Chan et al., 2021, etc). The unified treatments shed new lights on the sets of originally specialized methods and foster new progresses in the respective fields.

Integrating diverse sources of information in training has been explored in previous work which is often dedicated to specific tasks. Roth (2017) presented different ways of deriving supervision signals in different scenarios. Zhu et al. (2020) discussed the integration of physical and other knowledge in solving vision problems. The distant or weak supervision approaches (Mintz et al., 2009; Ratner et al., 2017) automatically create (noisy) instance labels from heuristics which are then used in the supervised training procedure. The panoramic learning we discussed here makes use of broader forms of experiences not necessarily amicable to be converted into supervised labels, such as reward, discriminator-like models, and many structured constraints. The experience function $f(\boldsymbol{y})$ offers such flexibility for expressing all those experiences.

9 Future Directions

We have presented a standardized machine learning formalism, materialized as the standard equation of the objective function, that formulates a vast algorithmic space governed by a few components regarding the experiences, model fitness measured with divergence, and uncertainty. The formalism gives a holistic view of the diverse landscape of learning paradigms, allows a mechanic way of designing ML solutions to new problems, and provides a vehicle towards panoramic learning that integrates all available experiences in building an AI agent. The work shapes a range of exciting open questions and opportunities for future study. We discuss a few of these directions in the below.

Learning in dynamic environments Our discussion on the standard equation mostly focused on learning in *static* environments (e.g., fixed data or reward distributions). A natural next step is to study the connection to other learning contexts where the tasks or data distributions are changing over time, sometimes with unknown dynamics, such as online and lifelong learning, to gain a new dimension of understanding about learning in *dynamic* environments. To this end, it is perhaps necessary to extend the definition of experience function to be time dependent. As how the current standard equation is used for solution design (Section 7), a standardized formalism of the broader ML landscape is expected to unleash even more power by enabling principled design of learning systems that continuously improve by interacting with and collecting diverse signals from the outer world.

Theoretical analysis of panoramic learning The paradigm of panoramic learning poses new questions about theoretical understanding. A question of particular importance in practice is about how we can guarantee better performance after integrating more experiences. The analysis is challenging because the different types of experiences can each encode different information, sometimes noisy and even conflicting with each other (e.g., not all data instances would comply with a logic rule), and thus plugging in an additional source of experience does not necessarily lead to positive effects. But before that, a more basic question to ask is perhaps about how we can characterize learning with some special or novel forms of experiences, such as logic rules and auxiliary models? What mathematical tools we may use for characterization, and what would the convergence guarantees, complexity, robustness, and other theoretical and statistical properties be? Inspired by how we generalized the specialized algorithms to new problems, a promising way of the theoretical analysis would be to again leverage the standard equation and repurpose the existing analyses originally dealing with supervised learning, online learning, and reinforcement learning, to now analyze the learning process with all other experiences.

From standardization to automation As in other mature engineering disciplines such as mechanical engineering, standardization is followed by automation. The standardized ML formalism opens up the possibility of automating the process of creating and improving ML algorithms and solutions. The current "AutoML" practice has largely focused on automatic search of neural network architectures and hyperparameters, thanks to the well-defined architectural and hyperparameter search spaces. The standard equation that defines a structured algorithmic space would similarly suggest opportunities for automatic search or optimization of learning algorithms, which is expected to be more efficient than direct search on the programming code space (Real et al., 2020). We have briefly discussed in Section 7 how new algorithms can mechanically be created by composing experiences and/or other algorithmic components together. It would be significant to have an automated engine that further streamlines the process. For example, once a new advance is made to reinforcement learning, the engine would automatically amplify the progress and deliver enhanced functionalities of learning data manipulation and adapting knowledge constraints. Similarly, domain experts can simply input a variety of experiences available in their own problem, and expect an algorithm to be automatically composed to learn the target model they want. The sophisticated algorithm manipulation and creation would greatly simplify machine learning workflow in practice and boost the accessibility of ML to much broader users.

From Maxwell's equations to general relativity, and to quantum mechanics and standard model, "Physics is the study of symmetry." remarked physicist Phil Anderson. The end goal of physics research seems to be clear—a "theory of everything" that fully explains and links together all physical aspects. The "end goal" of ML/AI is surely much elusive. Yet the unifying way of thinking would be incredibly valuable to continuously unleash the extensive more power of the current vibrant research, to produce more principled understanding, and to build more versatile AI solutions.

References

A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. Riedmiller. Maximum a posteriori policy optimisation. In ICLR, 2018.

Y. Altun and A. Smola. Unifying divergence minimization and statistical inference via convex duality. In International Conference on Computational Learning Theory, pages 139–153. Springer, 2006.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. 2017.

S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. Theory of Computing, 8(1):121–164, 2012.

S. H. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss Markov random fields and probabilistic soft logic. The Journal of Machine Learning Research, 18(1):3846–3912, 2017.

S. H. Bach, D. Rodriguez, Y. Liu, C. Luo, H. Shao, C. Xia, S. Sen, A. Ratner, B. Hancock, H. Alborzi, et al. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In Proceedings of the 2019 International Conference on Management of Data, pages 362–375, 2019.

H. H. Bauschke and J. M. Borwein. On projection algorithms for solving convex feasibility problems. SIAM review, 38(3):367–426, 1996.

C. M. Bishop. Model-based machine learning. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371(1984):20120222, 2013.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. arXiv preprint arXiv:2005.14165, 2020.

K. H. R. Chan, Y. Yu, C. You, H. Qi, J. Wright, and Y. Ma. ReduNet: A white-box deep network from the principle of maximizing rate reduction. arXiv preprint arXiv:2105.10446, 2021.

C. Chu, J. Blanchet, and P. Glynn. Probability functional descent: A unifying perspective on gans, variational inference, and reinforcement learning. In Proceedings of the 36th International Conference on Machine Learning, volume 97, 2019.

I. Csiszár. I-divergence geometry of probability distributions and minimization problems. The annals of probability, pages 146–158, 1975.

I. Dagan and S. P. Engelson. Committee-based sampling for training probabilistic classifiers. In Machine Learning Proceedings, pages 150–157. Elsevier, 1995.

P. Dayan and G. E. Hinton. Using expectation-maximization for reinforcement learning. Neural Computation, 9(2):271–278, 1997.

M. P. Deisenroth, G. Neumann, J. Peters, et al. A survey on policy search for robotics. Foundations and Trends® in Robotics, 2(1–2):1–142, 2013.

W. E. Deming and F. F. Stephan. On a least squares adjustment of a sampled frequency table when the expected marginal totals are known. The Annals of Mathematical Statistics, 11(4):427–444, 1940.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B (Methodological), 39(1):1–22, 1977.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In NAACL, 2019.

P. Domingos. The master algorithm: How the quest for the ultimate learning machine will remake our world. Basic Books, 2015.

S. Ertekin, J. Huang, L. Bottou, and L. Giles. Learning on the border: active learning in imbalanced data classification. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pages 127–136, 2007.

F. Farnia and D. Tse. A convex duality framework for gans. Advances in Neural Information Processing Systems, 31:5248–5258, 2018.

L. T. Fernholz. Von Mises calculus for statistical functionals, volume 19. Springer Science & Business Media, 2012.

K. Ganchev, J. Gillenwater, B. Taskar, et al. Posterior regularization for structured latent variable models. Journal of Machine Learning Research, 11(Jul):2001–2049, 2010.

I. D. Gebru, X. Alameda-Pineda, F. Forbes, and R. Horaud. EM algorithms for weighted-data clustering with application to audio-visual scene analysis. IEEE transactions on pattern analysis and machine intelligence, 38(12):2402–2415, 2016.

P. W. Glynn. Likelihood ratio gradient estimation for stochastic systems. Communications of the ACM, 33 (10):75–84, 1990.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NeurIPS, pages 2672–2680, 2014.

M. Gori. Machine Learning: A constraint-based approach. Morgan Kaufmann, 2017.

M. U. Gutmann, R. Dutta, S. Kaski, and J. Corander. Statistical inference of intractable generative models via classification. arXiv preprint arXiv:1407.4981, 2014.

G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The" wake-sleep" algorithm for unsupervised neural networks. Science, 268(5214):1158, 1995.

R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. VIME: Variational information maximizing exploration. In NeurIPS, 2016.

Z. Hu, X. Ma, Z. Liu, E. Hovy, and E. Xing. Harnessing deep neural networks with logic rules. In ACL, 2016.

Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. Toward controlled generation of text. In ICML, 2017.

Z. Hu, Z. Yang, R. Salakhutdinov, X. Liang, L. Qin, H. Dong, and E. Xing. Deep generative models with learnable knowledge constraints. In NeurIPS, 2018.

Z. Hu, B. Tan, R. Salakhutdinov, T. Mitchell, and E. Xing. Learning data manipulation for augmentation and weighting. In NeurIPS, 2019a.

Z. Hu, A. G. Wilson, C. Finn, L. Lee, W. Neiswanger, L. Qin, T. Berg-Kirkpatrick, R. Salakhutdinov, and E. P. Xing. The NeurIPS wokrshop on learning with rich experience: Integration of learning paradigms. 2019b. URL https://sites.google.com/view/neurips2019lire.

T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In Advances in neural information processing systems, pages 470–476, 2000.

E. Jaynes. Comment. The American Statistician, 42(4):280–281, 1988.

E. T. Jaynes. Information theory and statistical mechanics. Physical review, 106(4):620, 1957.

M. I. Jordan. An introduction to probabilistic graphical models, 2003.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. Machine learning, 37(2):183–233, 1999.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In ICLR, 2014.

J. Knoblauch, J. Jewson, and T. Damoulas. Generalized variational inference: Three arguments for deriving new posteriors. arXiv preprint arXiv:1904.02063, 2019.

P. Langley. Toward a unified science of machine learning. Machine Learning, 1989.

Y. Lecun and I. Misra. Self-supervised learning: The dark matter of intelligence. https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence, 2021.

S. Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv preprint arXiv:1805.00909, 2018.

L. Li, M. L. Littman, T. J. Walsh, and A. L. Strehl. Knows what it knows: a framework for self-aware learning. Machine learning, 2011.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pages 1003–1011, 2009.

A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. arXiv preprint arXiv:1402.0030, 2014.

S. Mohamed and B. Lakshminarayanan. Learning in implicit generative models. arXiv preprint arXiv:1610.03483, 2016.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In Learning in graphical models, pages 355–368. Springer, 1998.

M. Norouzi, S. Bengio, N. Jaitly, M. Schuster, Y. Wu, D. Schuurmans, et al. Reward augmented maximum likelihood for neural structured prediction. In Advances In Neural Information Processing Systems, pages 1723–1731, 2016.

S. Nowozin, B. Cseke, and R. Tomioka. f-GAN: Training generative neural samplers using variational divergence minimization. In NeurIPS, pages 271–279, 2016.

J. Paisley, D. Blei, and M. Jordan. Variational Bayesian inference with stochastic search. In ICML, 2012.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In ICML, 2017.

D. Pathak, D. Gandhi, and A. Gupta. Self-supervised exploration via disagreement. In ICML, 2019.

R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In Artificial Intelligence and Statistics, pages 814–822, 2014.

A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. Snorkel: Rapid training data creation with weak supervision. In Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases, volume 11, page 269. NIH Public Access, 2017.

K. Rawlik, M. Toussaint, and S. Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. In IJCAI, 2013.

E. Real, C. Liang, D. So, and Q. Le. AutoML-zero: evolving machine learning algorithms from scratch. In International Conference on Machine Learning, pages 8007–8019. PMLR, 2020.

D. Roth. Incidental supervision: Moving beyond supervised learning. In Thirty-First AAAI Conference on Artificial Intelligence, 2017.

S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. Neural computation, 11(2): 305–345, 1999.

R. Samdani, M.-W. Chang, and D. Roth. Unified expectation maximization. In ACL, 2012.

F. Santambrogio. Optimal transport for applied mathematicians. Birkäuser, NY, 55(58-63):94, 2015.

J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). IEEE Transactions on Autonomous Mental Development, 2010.

B. Settles. Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, 6(1):1–114, 2012.

T. Shen, T. Lei, R. Barzilay, and T. Jaakkola. Style transfer from non-parallel text by cross-alignment. In NeurIPS, 2017.

S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg. Intrinsically motivated reinforcement learning: An evolutionary perspective. IEEE Transactions on Autonomous Mental Development, 2010.

R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction (2nd edition). 2017.

R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In Advances in neural information processing systems, pages 1057–1063, 2000.

B. Tan, L. Qin, E. Xing, and Z. Hu. Summarizing text on any aspects: A knowledge-informed weakly-supervised approach. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6301–6309, 2020.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In Advances in neural information processing systems, pages 25–32, 2004.

Y. W. Teh and M. Welling. On improving the efficiency of the iterative proportional fitting procedure. In AIStats, 2003.

V. N. Vapnik. Statistical learning theory. John Wiley and Sons, 1998.

M. J. Wainwright and M. I. Jordan. A variational principle for graphical models. 2005.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, pages 1–305, 2008.

Y. Wu, P. Zhou, A. G. Wilson, E. P. Xing, and Z. Hu. Improving GAN training with probability ratio clipping and sample reweighting. In NeurIPS, 2020.

Y. N. Wu, R. Gao, T. Han, and S.-C. Zhu. A tale of three probabilistic families: Discriminative, descriptive, and generative models. Quarterly of Applied Mathematics, 77(2):423–465, 2019.

E. P. Xing, M. I. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. In Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence, pages 583–591, 2002.

Z. Yang, Z. Hu, C. Dyer, E. Xing, and T. Berg-Kirkpatrick. Unsupervised text style transfer using language models as discriminators. In NeurIPS, 2018.

J. Yu, M.-S. Yang, and E. S. Lee. Sample-weighted clustering methods. Computers & mathematics with applications, 62(5):2200–2208, 2011.

A. Zellner. Optimal information processing and bayes's theorem. The American Statistician, 42(4):278–280, 1988.

Z. Zheng, J. Oh, and S. Singh. On learning intrinsic rewards for policy gradient methods. In NeurIPS, 2018.

J. Zhu and E. P. Xing. Maximum entropy discrimination markov networks. Journal of Machine Learning Research, 10(11), 2009.

J. Zhu, N. Chen, and E. P. Xing. Bayesian inference with posterior regularization and applications to infinite latent SVMs. JMLR, 15(1):1799–1847, 2014.

Y. Zhu, T. Gao, L. Fan, S. Huang, M. Edmonds, H. Liu, F. Gao, C. Zhang, S. Qi, Y. N. Wu, et al. Dark, beyond deep: A paradigm shift to cognitive ai with humanlike common sense. Engineering, 6(3):310–345, 2020.

B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In AAAI, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.